UNIVERSITÉ DE RENNES 1

Projet Magistère Statistique et Modélisation économique

Application de la théorie des graphes à un exemple concret : les achats communs sur Amazon

Guillaume LE FLOCH
Année 2017-2018

Encadrante : Mme Véronique THELEN

Table des matières

1 Introduction et problématique

Dans ce projet, nous allons étudier les liens entre certains produits achetés sur **Amazon** à partir du jeu de données *Amazon0601.txt* qui contient des associations de produits fréquemment achetés ensemble. Ces données ont été collectées le 1^{er} juin 2003, et sont construites de la façon suivante :

- si un produit *i* est fréquemment acheté avec un produit *j*, le graphe contiendra alors un lien allant de *i* vers *j*. C'est donc un graphe orienté.
- le graphe contient 403394 nœuds et 3387388 liens.

En considérant ces chiffres, on comprend donc qu'on ne pourra pas représenter les données ici. Ne disposant pas de la puissance calculatoire suffisante, il ne sera également pas possible ici de travailler sur tous les produits : nous allons donc nous focaliser sur un sous-ensemble contenant les DVD uniquement. Ce sous-ensemble représente 4442 nœuds ce qui donne une représentation encore assez dense du réseau.

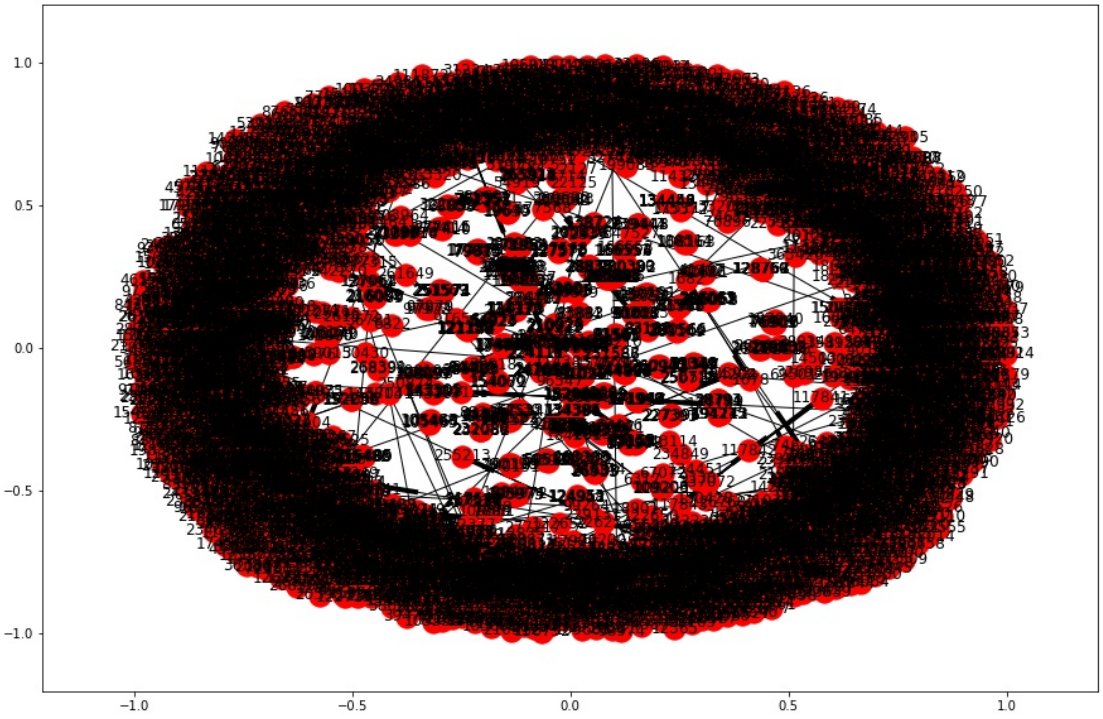

Figure 1 – Une telle représentation ne permet évidemment pas de tirer des conclusions

Cependant, grâce à l'analyse des caractéristiques de ce réseau d'achats nous allons pouvoir répondre à la problématique suivante : **identifier les produits et associations de produits phares dans le but de faire de la recommandation client**. En effet, si on arrive à identifier quels produits sont **au centre** du réseau d'achat, on sera en mesure premièrement de mettre l'accent sur ces produits au niveau publicitaire, mais également de focaliser les recommandations sur les produits auxquels ils sont associés, et dans un dernier temps de pouvoir effectuer de la prédiction sur des associations qui n'existent pas encore mais qui pourraient intéresser les acheteurs. Tout ceci se base bien évidemment sur le concept d'homophilie : on émet l'hypothèse que les acheteurs ayant des comportements d'achats similaires vont être intéressés par le même type de produits.

Avant de passer à l'analyse des caractéristiques de ce réseau d'achats, nous allons revenir sur certaines notions et sur le vocabulaire lié aux graphes.

2 Quelques rappels de vocabulaire

Nous allons commencer cette partie en rappelant quelques définitions élémentaires et en les illustrant.

- on définit un **graphe** comme étant un ensemble de points et de lignes reliant ces points
- ces points sont appelés communément des **nœuds** et sont reliés par des **arcs**
- le graphe est orienté si l'association entre deux nœuds se fait dans un sens mais pas dans l'autre (présence de flèches sur le graphe pour représenter le sens)

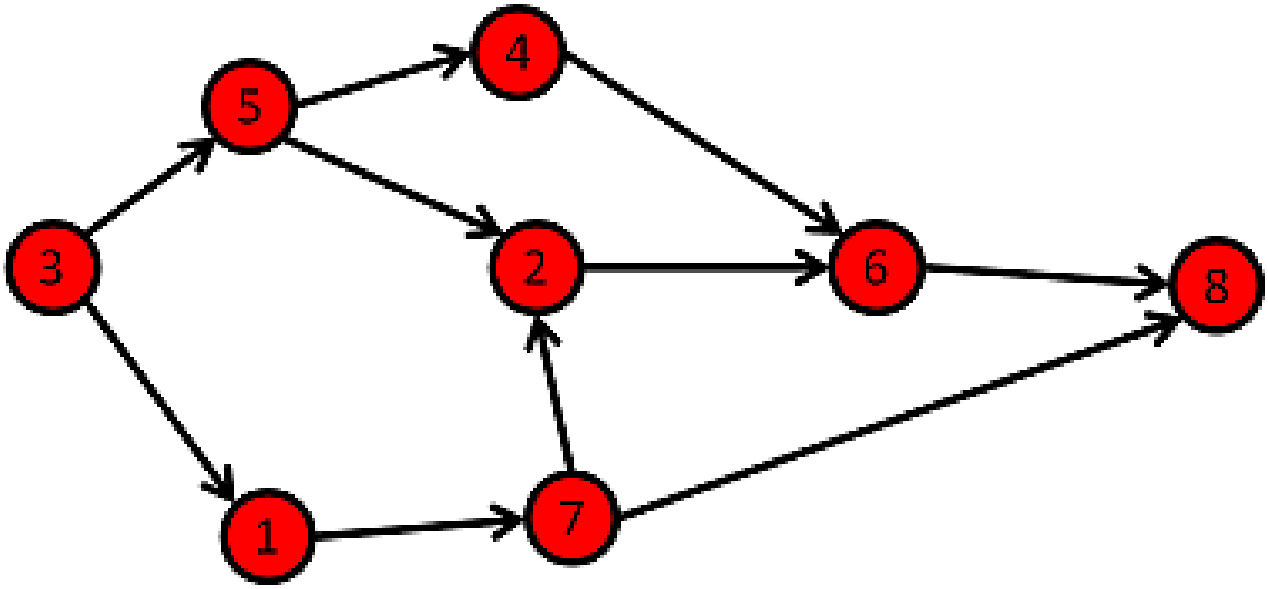

FIGURE 2 – Exemple de nœuds et d'arcs au sein d'un graphe orienté

De façon mathématique, on va noter un graphe comme étant (N, g) avec N représentant le nombre d'agents, qui dans notre étude sont les produits achetés sur **Amazon** (ou encore les n nœuds du graphe), et g une matrice d'adjacence de taille n x n telle que $g = [g_{ij}]$ avec $i, j \in N$. Dans ce projet, nous étudions un graphe orienté, ainsi g_{ij} se définit de la façon suivante :

$$g_{ij} = \begin{cases} 1 & \text{si } i \to j \\ 0 & \text{sinon.} \end{cases}$$

La somme sur chaque colonne j de la matrice nous donnera donc le nombre de degrés entrants du nœud j en question. D'autres notions dont nous allons avoir besoin par la suite pour analyser notre graphe orienté sont les suivantes :

- Une **séquence** est une suite consécutive d'arcs $\{i_1, i_2\}, \{i_2, i_3\}, \ldots, \{i_{K-1}, i_K\}$
- Un **chemin** entre le nœud i et j est une séquence d'arcs $\{i_1, i_2\}, \{i_2, i_3\}, \ldots, \{i_{K-1}, i_K\}$ tel que $i_1 = i$ et $i_K = j$ et chaque nœud dans la séquence est distinct
- Un **cycle** est un chemin avec un arc final revenant au nœud initial
- Une **distance géodésique** entre les nœuds i et j est le plus court chemin (avec le nombre minimum de nœuds) entre les deux nœuds. On notera $I(i, j)$ cette mesure

Après avoir effectué ces quelques rappels nous pouvons donc nous diriger vers l'analyse des caractéristiques du réseau.

3 Analyse des caractéristiques du réseau

L'enjeu de cette partie va être de réussir à dégager les grandes tendances du réseau, de découvrir quels produits se trouvent au **centre** du réseau. Pour cela nous allons donc étudier plusieurs mesures de **centralité** ou encore « **d'importance** », et tenter d'obtenir un *Top 30* des produits les plus populaires. Le but de cette manœuvre sera de faire ressortir les produits les plus importants, qui ont rencontré le plus de succès au niveau des achats, afin de les utiliser dans la dernière partie (la prédiction) et d'optimiser la stratégie marketing d'Amazon par conséquent, en proposant de nouvelles recommandations autour de ces produits. Cela permettra de donner plus de visibilité à des produits potentiellement intéressants, qui en manquent pour le moment. Le choix des 30 premiers produits est bien évidemment arbitraire, on pourrait regarder plus de produits mais on ne retiendra que cet échantillon qui nous permettra déjà de comprendre le principe.

3.1 Degré de centralité

Dans le cas d'un graphe orienté, le degré de centralité (ou centralité de degré) se définit comme étant le nombre de degrés entrants d'un nœud, c'est-à-dire que l'on va compter le nombre de liens qui pointent vers chaque nœud. De façon mathématique, cela correspond tout simplement pour un nœud i à $\sum_j g_{ij}$.

La fonction *in_degree_centrality* du module *NetworkX* de Python renvoie, pour chaque nœud, une mesure différente puisqu'elle est normalisée en divisant par le degré de centralité maximum possible dans un graphe à N-1 nœuds. Cela n'a pas d'importance ici puisque l'on cherche simplement à classer les produits grâce à cette mesure, mais il est toutefois important de noter cette spécificité afin de mieux comprendre les résultats que nous renvoie le programme. Les résultats sont donc les suivants :

title	score
Shapes	0.003378
Cujo	0.003152
Royal	0.002927
Plays	0.002702
Pie?	0.002252
Traffic	0.002027
Stalker	0.001801
Scorpion	0.001801
Claudel	0.001801
Sweethearts	0.001801
Deuces/Utopia	0.001576
Draw	0.001576
Night	0.001576
Retrospective	0.001576
Home	0.001576
Dinosaur	0.001351
Year-90's	0.001351
Regeneration	0.001351
Tramp	0.001351
Betty	0.001351
Women	0.001351
Hellion	0.001351
Hole	0.001351
Collection	0.001351
Visiting	0.001351
Man	0.001351
Feast	0.001351
Showtime	0.001351
Values	0.001351
London	0.001351

FIGURE 3 – Top 30 des produits selon le nombre de degrés entrant

Nous allons maintenant nous intéresser à une mesure qui prend plutôt en compte la notion de **proximité** et de **distance** plutôt que la « popularité ». Cela va donc nous donner les produits les plus « homogènes ».

3.2 Centralité de proximité

Ici on va plutôt faire ressortir les produits qui sont les plus « passe-partout » dans le sens où ils sont les plus neutres, ce ne sont pas des extrêmes. Cela décrit donc la centralité de proximité, qui fait appelle à la notion de distance géodésique présentée en introduction. Dans le cas d'un graphe orienté il est important de préciser que les distances **vers** un nœud sont considérées comme une mesure plus significative de centralité, puisqu'un nœud a peu de contrôle sur ses liens entrants (c'est le cas dans notre exemple). Cette mesure se définit donc de la façon suivante :

$$C_{closeness}(i) = \frac{n-1}{\sum_{j=1}^{n-1} I(i,j)}$$

Les résultats associés à cette mesure sont les suivants :

title	score
Claudel	0.001658
Mermaid	0.001658
Swope	0.001601
Tale	0.00152
Pleasure	0.001441
Night	0.001403
Discovery	0.001351
Flexibility	0.001351
Burning	0.001351
Australia	0.001351
Mines	0.001351
Values	0.001351
Feast	0.001303
H	0.001303
Scorpion	0.001303
Change	0.001216
Unsatisfied	0.001216
Road	0.001216
Married	0.001216
Miles	0.001158
Head	0.001158
Land	0.001158
Coming	0.001158
Lawman	0.001158
Bride	0.001158
Visiting	0.001158
Lyon	0.001158
Odyssey	0.001158
Deluxe	0.001158
Music	0.001158

FIGURE 4 – Top 30 des produits selon la centralité de proximité

On peut d'ores et déjà constater que le classement est différent de celui renvoyé par la première mesure, ce qui n'est pas surprenant puisqu'on ne mesure pas la même chose. Nous allons à présent nous intéresser à une notion de centralité encore différente : la **centralité d'intermédiarité**.

3.3 Centralité d'intermédiarité

Concernant cette nouvelle mesure, l'idée n'est plus de regarder la proximité ou encore distance aux autres nœuds du graphe mais plutôt de compter le nombre de fois où un nœud agit comme un point de passage le long du plus court chemin entre deux autres nœuds (distance géodésique). On définit donc cette mesure comme suit :

$$C_{betweenness}(k) = \frac{P_k(i,j)}{P(i,j)}$$

avec :

- $P_k(i,j)$ le nombre de distances géodésiques entre i et j que le nœud k lie

- $P(i, j)$ le nombre de distances géodésiques entre i et j

On peut renormaliser cette quantité par $\frac{1}{(n-1)(n-2)}$ dans le cas des graphes orientés et c'est d'ailleurs ce que fait par défaut la fonction *betweenness_centrality* en Python. Encore une fois, on obtient un classement différent...

title	score
Swope	0.000002
Retrospective	0.000002
Hook-Up	0.000001
Values	0.000001
Pleasure	0.000001
Tune-Up	0.000001
Thighs	0.000001
Dream	0.000001
Mermaid	0.000001
Claudel	0.000001
Hakaider	0.000001
Edition	0.000001
Core	0.000001
Job	0.000001
Montreal	0.000001
Tramp	0.000001
Traffic	0.000001
Grist	0.000001
Possessed	0.000001
Serendipity	0.000001
Keoma	0.000001
Seaward	0.000001
Series	0.000001
Sons	0.000001
Sweethearts	0.000001
Visiting	0.000001
Vacations	0.000001
Cujo	0.000001
Compton	0.000001
Shoulder	0.000001

FIGURE 5 – Top 30 des produits selon la centralité d'intermédiarité

On pourrait par exemple utiliser ces résultats quand on regarde des parcours de navigation pour identifier les produits qui se retrouvent fréquemment sur le « chemin » entre 2 produits. Dans la suite, nous allons utiliser deux autres mesures qui, contrairement aux précédentes, sont moins visuelles et plus abstraites.

3.4 Centralité de vecteur propre

Nous en venons donc à un quatrième indicateur appelé **centralité de vecteur propre** (ou encore centralité spectrale). Comme son nom l'indique, cette mesure découle de la notion de vecteur propre. Elle nous permet de mesurer l'influence et d'attribuer des scores relatifs à tous les nœuds d'un réseau. Pourquoi ? Si l'on note C_i cette mesure de centralité pour le nœud i, alors :

$$C_i \text{ est proportionnelle à } \sum_{j \ voisin \ de \ i} C_j \Leftrightarrow C_i = \lambda \sum_j g_{ij} C_j$$
(ou encore $C = \lambda g C$ avec g matrice d'adjacence du graphe G)

On compte ainsi les centralités des nœuds j avec lesquels le nœud i est connecté, ce qui donne un système d'équations et d'inconnues : d'où la notion de vecteur propre. D'un point de vue mathématique, il existe en général plusieurs valeurs propres et vecteurs propres solutions. Cependant, en vertu du théorème de *Perron-Frobenius*, si l'on impose que les coefficients du vecteur propre doivent être positifs, alors il existe une unique solution pour la mesure de centralité souhaitée, qui nous est fournie par la plus grande valeur propre λ. Pour cette mesure, les résultats donnent encore une fois un classement différents et sont à la page suivante.

title	score
Discovery	0.404544
Flexibility	0.404544
Burning	0.404544
Australia	0.404544
Odyssey	0.359331
Lyon	0.359331
Encounter	0.294037
Claudel	0.011975
Scorpion	0.011975
Feast	0.010263
Night	0.009317
Change	0.008816
H	0.00624
Spring	0.005563
Mermaid	0.005246
Tale	0.005084
Ponette	0.001005
Horse	2.2e-05
Coming	2.2e-05
Head	2.2e-05
Mines	2.2e-05
Lawman	1.9e-05
Land	1.9e-05
Under	0.000005
Draw	0.0
Home	0.0
Beanstalk	0.0
Trail	0.0
9th December	0.0
Gorilla	0.0

FIGURE 6 – Top 30 des produits selon la centralité de vecteur propre

On peut donc remarquer que certains scores sont très faibles et même nuls pour certains, cela vient de la nature même de la mesure puisque ce sont des valeurs propres. On obtient ces scores à partir de la fonction *eigenvector_centrality* en Python. Il est à noter que pour le cas d'un graphe orienté, cette fonction utilise la « left » *eigenvector centrality* dans le sens où les voisins considérés pour un nœud j sont les nœuds i qui pointent vers lui. L'aide de la fonction nous précise également que le calcul des vecteurs propres se fait par la méthode **power iteration** sur laquelle se base également la dernière mesure que nous allons utiliser : la méthode **PageRank**. Nous allons donc détailler tout cela dans la prochaine partie.

3.5 L'algorithme PageRank de Google

Historiquement, cette méthode était utilisée pour classer les sites web par domaines, et désormais c'est l'algorithme utilisé par les moteurs de recherche. Le principe est de calculer l'importance d'une page (d'un nœud) en étudiant ses liens (entrants et sortants) dans le graphe. On retrouve deux idées derrière cette méthode :

- Donner de l'importance aux pages souvent citées
- Donner de l'importance aux pages citées par des pages importantes (on retrouve la notion de prestige des voisins)

On peut donc s'inspirer de cette méthode en la transposant à notre problème, les pages web devenant les différents produits. Si l'on considère les liens entrants comme des votes :

- Le poids d'un vote est proportionnel à l'importance du nœud source
- Le poids d'un vote est inversement proportionnel au nombre de votes de son nœud source
- L'importance d'un nœud est égale à la somme des poids des votes vers ce nœud

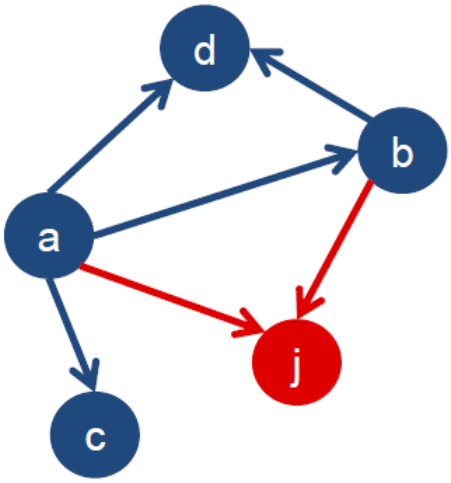

FIGURE 7 – Illustration des principes énoncés précédemment : $r_j = \frac{r_a}{4} + \frac{r_b}{2}$

En généralisant, on arrive donc au système d'équations :

$$\forall j, r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

En ajoutant la contrainte suivante sur les poids :

$$\sum_i r_i = 1$$

Et en notant la matrice M stochastique :

$$M_{ij} = \begin{cases} \frac{1}{d_i} & \text{si } i \to j \\ 0 & \text{sinon.} \end{cases}$$

On cherche r tel que $r = Mr$. On en revient encore une fois au problème de trouver un vecteur propre. Comme pour la centralité de vecteur propre, le calcul se fait à partir de l'algorithme **power iteration** qui s'exprime de la façon suivante :

Algorithm 1 Algorithme power iteration

Entrées :
Matrice de transitions M
$t = 1$
$r(t) = \begin{pmatrix} \frac{1}{n} \\ \frac{1}{n} \\ \vdots \\ \frac{1}{n} \end{pmatrix}$
Tant que Pas de convergence de l'algorithme (*i.e* : r(t) et r(t-1) dissimilaires)
$t = t + 1$
Pour j = 1,...,n
$r_j(t) = 0$
Pour i prédecesseur de j
$r_j(t)$ += $\frac{r_i(t-1)}{d_i}$
Fin pour
Fin pour
Fin du « Tant que »
Retourner Vecteur $r(t)$

La preuve de convergence de cet algorithme est issue des processus de marches aléatoires (un graphe peut être vu comme une chaîne de Markov). Pour cela il faut que M soit :

- **Stochastique** : les lignes somment à 1 (matrice de transition d'une chaîne de Markov)
- **Irréductible** : il existe une probabilité non nulle d'aller d'un nœud quelconque i à un autre nœud quelconque j : $M_{i,j} > 0$
- **Apériodique** : il n'existe pas (k, i) tels que l'intervalle entre deux visites de i est **toujours** un multiple de k

Si ce n'est pas respecté, on peut se retrouver en présence de 2 cas problématiques :

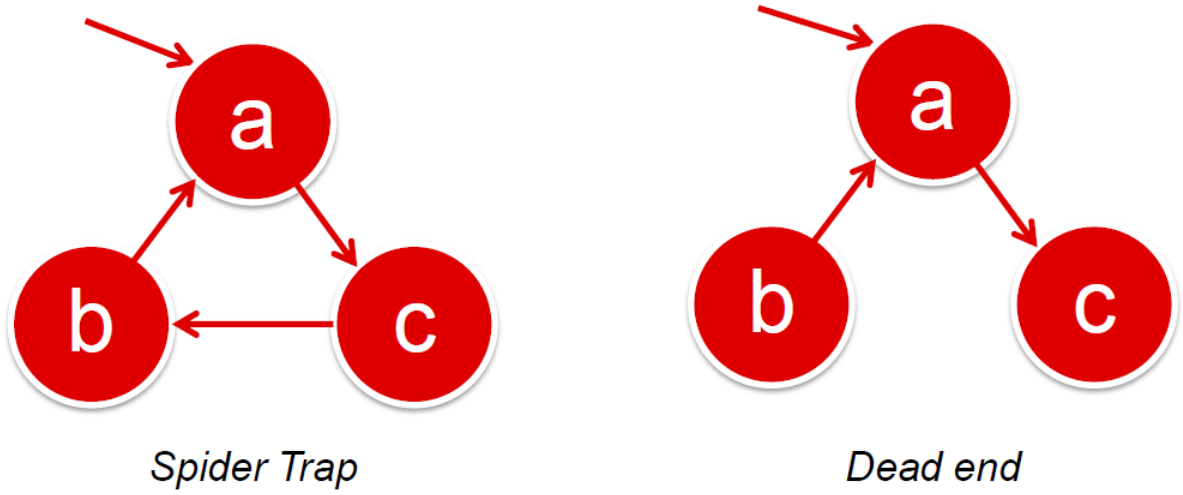

FIGURE 8 – Ces 2 types de configuration posent problème

Que l'on peut résoudre ainsi en informatique :

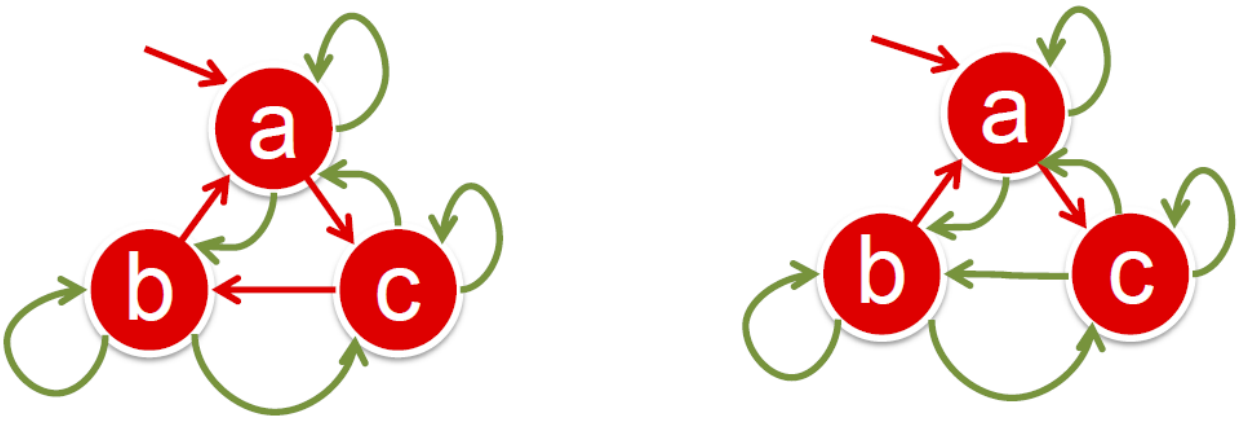

FIGURE 9 – On autorise la téléportation pour résoudre le problème de blocage

La fonction *pagerank* implémentée dans la librairie *NetworkX* de python utilise la formulation de **Google** pour effectuer les calculs :

$$r_j = \sum_{i \to j} \alpha \frac{r_i}{d_i} + (1 - \alpha)\frac{1}{n}$$

Les valeurs typiques pour α se situent entre 0.8 et 0.9. Pour cette raison nous alons fixer $\alpha = 0.85$ et laisser le nombre d'itérations maximum de l'algorithme **power iteration** par défaut (égal à 100). L'algorithme a bien convergé et les résultats sont les suivants :

title	score
Cujo	0.002514
Live	0.002204
Traffic	0.001929
Pie?	0.001928
Betty	0.001823
Sweethearts	0.00182
Movies	0.001664
Collection	0.001213
Concert	0.001171
Live	0.001166
Collection	0.001083
Retrospective	0.001053
Hole	0.000998
Dragon	0.000956
Miles	0.000956
Extraordinaire	0.000953
Dragon	0.000946
Montreal	0.00093
Orchestra	0.000921
Shirt	0.000918
Racing	0.000915
Sennett	0.000902
City	0.000879
Edition	0.000862
Necklace	0.000851
Ground	0.00083
You	0.0008
Sons	0.000792
Flower	0.000786
Night	0.000756

FIGURE 10 – Top 30 des produits selon l'algorithme PageRank

D'autres indicateurs auraient pu être essayés, tels que la **centralité de Katz** par exemple, mais l'idée de cette mesure repose sur la notion de prestige des voisins d'un nœud et dans cette étude on n'a pas d'informations sur le prestige des DVD. Pour cette raison, il n'a donc pas été jugé nécessaire d'utiliser cette mesure.

3.6 Combinaison des différents indicateurs

Après avoir regardé ces différentes méthodes, on s'aperçoit que les résultats changent à chaque fois, du fait de la nature même de chaque mesure. Une question que l'on pourrait se poser serait « Quelle est la meilleure ? ». Cependant c'est comme lorsqu'on choisit un modèle statistique, on effectue notre choix selon un **critère**. Si l'on avait un critère particulier à respecter, en fonction de la définition mathématique de chaque méthode ou algorithme on pourrait donc en privilégier une. Dans cette étude, l'approche va être différente puisque l'on va combiner les résultats. On ne va pas accorder plus d'importance à une méthode qu'à une autre, la procédure est la suivante :

- On normalise les scores de chaque produit pour chaque méthode par rapport au score maximum (fonction *MaxAbsScaler* en Python)
- On additionne les scores normalisés de chaque produit pour obtenir un score cumulé des méthodes

Cette procédure est totalement arbitraire et ne devra pas forcément être utilisée dans tous les cas, elle va nous servir simplement dans le cadre de ce projet à résumer l'information renvoyée par chaque indicateur de centralité/importance. On va donc classer les produits selon un critère « d'homogénéité », dans le sens où ce sont les produits les plus « complets » au sens de la combinaison des différents indicateurs étudiés, qui seront désignés comme étant les meilleurs.

Le classement des produits renvoyé par cette manœuvre est le suivant :

title	score
Cujo	2.526279
Swope	2.39432
Burning	2.377629
Australia	2.377629
Flexibility	2.377629
Discovery	2.377629
Claudel	2.207304
Sweethearts	2.071748
Retrospective	2.048318
Odyssey	2.033815
Lyon	2.033815
Values	1.890345
Traffic	1.874581
Visiting	1.759769
Pie?	1.740892
Mermaid	1.739523
Scorpion	1.720533
Night	1.710866
Vacations	1.704715
Encounter	1.699096
Hook-Up	1.629231
Pleasure	1.573471
Edition	1.572943
Mines	1.545818
Possessed	1.540364
Matrix	1.474924
Morrison	1.463393
Keoma	1.459257
Feast	1.436367
Draw	1.426822

FIGURE 11 – Top 30 des produits avec la combinaison des méthodes

En fonction des critères que nous nous sommes fixés, nous avons à présent identifié les DVD centraux dans le graphe. Nous allons désormais pouvoir passer à la dernière partie de cette étude qui concerne la prédiction de liens. Pour cela nous allons présenter la méthode choisie et les résultats qu'elle renvoie sur quelques un des premiers DVD de la liste établie.

4 Prédiction de liens

On aurait pu être tenté d'arrêter l'étude à la simple identification des nœuds centraux du graphe et ensuite pour chacun d'entre eux, d'effectuer des recommandations basées sur les voisins de chaque produit dans le graphe. Cependant, un réseau est une structure dynamique, qui est amenée à évoluer au fil du temps. Un des enjeux va donc être de pouvoir modéliser le comportement d'une telle structure afin de tenter de prédire la forme qu'elle aura par la suite. Plus particulièrement dans notre cas, nous nous intéressons aux produits qui ont été achetés ensemble. On travaille donc sur ce que l'on a observé, mais le but est d'enrichir l'analyse en essayant de définir quels produits pourraient également intéresser un acheteur, d'inférer sur de nouvelles combinaisons de produits qui n'existent pas encore. Pour cela, on va attribuer un score de « proximité » ou de similarité à chaque paire de nœuds (i, j).

Une façon d'y parvenir est d'utiliser l'index **Adamic-Adar**. Il existe évidemment d'autres méthodes de prédictions, mais dans cette étude nous n'allons utiliser que celle-là car les résultats sont assez similaires en général entre les différentes méthodes qui se basent sur le concept du voisinage d'un nœud. De manière mathématique, l'index **Adamic-Adar** se définit de la manière suivante pour deux nœuds u et v :

$$AA_{index}(u, v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{log|\Gamma(w)|}$$

où $\Gamma(u)$ désigne l'ensemble des voisins d'un nœud u.

Toutefois, comme le précise la documentation de *NetworkX*, la fonction *adamic_adar_index* n'est adaptée qu'aux graphes non-orientés. Pour utiliser cet indicateur il a donc fallu procéder à une adaptation de la fonction afin de la rendre compatible à notre problème (considérer les voisins, que le lien soit entrant ou sortant dans un graphe orienté). En utilisant cette méthode, voici donc quelques prédictions que l'on peut faire pour les produits que nous avons identifiés comme étant les plus importants. Pour le DVD **Cujo** identifié comme étant le n°1 de notre classement, on obtient les recommandations suivantes :

title	score
Dragon	1.442695
Derailed	0.910239
Access	0.621335
Collection	0.621335
Joshua	0.621335

On proposerait donc à un client en train de visionner/acheter le DVD **Cujo** de consulter également la page du DVD **Dragon** ou bien dans une moindre mesure **Derailed**. Encore une fois, les choix sont arbitraires sur le nombre de recommandations que l'on va juger pertinentes de faire. Ici on renvoie les 5 produits ayant obtenu l'index Adamic-Adar le plus élevé, mais il n'est pas forcément nécessaire d'aller aussi loin, ou bien au contraire on peut vouloir en renvoyer plus. Parfois, les résultats ne nous laissent pas le choix, comme par exemple pour le n°2 du classement, **Swope** :

title	score
Home	0.621335
Endgame	0.0
Stage	0.0
Cujo	0.0
Royal	0.0

Ici, on ne pourrait recommander que le DVD **Home**. On remarque également que l'index est plus faible entre **Swope** et **Home** (~ 0.62) que tout à l'heure entre **Cujo** et **Dragon** (~ 1.44). On recommanderait donc avec moins de confiance.

5 Bilan

Comme nous l'avons énoncé dans les différentes parties qui précèdent, il n'y a pas de vérité absolue dans les résultats qui sont renvoyés. Premièrement parce qu'ils découlent de choix arbitraires qui doivent être faits par l'utilisateur en fonction du type de problème auquel il fait face, ainsi que de critères pour choisir une mesure plutôt qu'une autre ou pour effectuer un classement des nœuds dans le graphe.

Il existe une multitude de mesures de centralité, d'importance pour les nœuds d'un graphe, et dans ce projet seule une petite partie a été utilisée afin d'illustrer le principe de centralité dans la théorie des graphes.
Il en va de même pour la prédiction. Ici, le choix de l'index Adamic-Adar a été fait, mais il existe d'autres mesures telles que les mesures de distance dans le graphe, les voisins communs, le coefficient de Jaccard ou encore la mesure de Katz. Toutes ces méthodes sont basées sur le concept de voisinage d'un nœud et renvoient donc des résultats similaires à ceux de l'index Adamic-Adar. C'est encore une fois un choix arbitraire.
On pourrait également être tenté de calculer les prédictions pour chaque nœud, et les classer afin d'obtenir les prédictions de liens les plus fortes entre 2 produits, plutôt que de renvoyer pour chaque article un classement des produits avec lesquels il a le plus de chance d'avoir un lien dans le futur. Nous avons vu que cela pouvait parfois poser problème. La stratégie de prédiction est ouverte, comme le reste, et devra donc être définie selon un ou des critères bien précis.

A titre personnel, ce projet m'aura permis d'avoir une première approche concrète d'application de la théorie des graphes à un problème de recommandations de produits. Cela me permet également d'appréhender et de me familiariser avec les outils permettant de répondre à ce type de problème, ce qui n'est pas négligeable étant donné que lors de mon stage de fin d'études au sein de **Cdiscount** je serai confronté à un problème d'optimisation des recommandations de produits par segment de clients. Grâce à ce projet, j'ai donc pu avoir une première approche, et quelques idées pour répondre à une partie du problème. Cependant, dans ce projet nous n'avons utilisé qu'un petit échantillon qui ne reflète pas du tout la taille des données manipulées en entreprise. Il faudra donc probablement adapter certaines méthodes qui ne pourront pas être utilisées telles quelles.

L'ensemble des codes ayant permis la réalisation de ce projet (extraction des produits d'intérêt, nettoyage des données, calcul des différents scores, écriture des résultats, ...) est disponible sur GitHub à l'adresse suivante :

https://github.com/guillaumelf/reco

6 Sources

- **Source des données** : *http ://snap.stanford.edu/data/amazon0601.html*

- **Documentation de NetworkX** : *https ://networkx.github.io/documentation/stable/reference/index.html*

- **J. Leskovec, L. Adamic and B. Adamic** : *The Dynamics of Viral Marketing. ACM Transactions on the Web (ACM TWEB), 1(1), 2007*