h e g

Recherche de nouvelles tendances sur Twitter avec le « data clustering »

Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES

par :

Alexis BECK

Conseiller au travail de Bachelor :

Alexandros KALOUSIS, Professeur HES

Genève, le 15 juillet 2013

Haute École de Gestion de Genève (HEG-GE)

Filière informatique de gestion

Hes·so GENÈVE
Haute Ecole Spécialisée de Suisse occidentale

Déclaration

Ce travail de Bachelor est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre de Bachelor en Informatique de Gestion. L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de Bachelor, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de Bachelor, du juré et de la HEG.

« J'atteste avoir réalisé seul le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève, le 15 juillet 2013

Alexis Beck

Remerciements

Mes premiers remerciements vont à Monsieur Alexandros Kalousis, mon directeur de mémoire, pour m'avoir conseillé et apporté son expérience tout au long de ce travail passionnant. Son cours de « data mining » m'a vivement intéressé et m'a donné envie de réaliser ce travail autour de son domaine.

Je remercie fortement Monsieur Phong Nguyen, assistant de recherche à l'université de Genève, pour son aide précieuse lorsque j'ai rencontré des difficultés et pour son suivi du projet.

Merci à Monsieur Nicolas Mayencourt, l'un des administrateurs des serveurs de l'université de Genève, pour avoir facilité mon travail sur les machines de calcul.

Mes parents, Aline Benoit et David Dos Reis Da Silva m'ont aidé à corriger la forme de ce travail, je les en remercie également.

Je terminerai par une dédicace spéciale à Monsieur Michel Kuhne, enseignant et véritable icône de la Haute Ecole de Gestion par son fameux cours d'algorithmique. C'est avec lui qu'est né le projet, mais le temps a manqué pour réaliser la partie de relativisation bayésienne des résultats que nous projetions.

Résumé

Ce mémoire est la description d'un projet de recherche qui a été réalisé en collaboration avec l'université de Genève. L'objectif est de découvrir de nouvelles tendances sur Twitter en appliquant des outils de « data mining ».

Twitter est le deuxième réseau social le plus important après Facebook. Il a des qualités intrinsèques intéressantes pour une analyse informatique ; chaque message publié est de 140 caractères maximum ce qui oblige les internautes à être concis dans leurs propos, le fait que la plupart des tweets soient publics est un autre avantage car leur récupération est gratuite. Bien que ses utilisateurs soient peu représentatifs d'une population globale, les nouveaux concepts, eux, transparaissent plus facilement sur le réseau social et c'est pour cela que cette voie d'utilisation a du sens.

L'analyse présentée ici s'intéresse au shampoing. Quels sont les dernières préoccupations des consommateurs quand ils parlent de ce sujet-là ? Pour répondre à cette question, nous avons créé des regroupements par un algorithme de « clustering » agglomératif. Connaître les concepts majeurs existants nous a permis ensuite de tenter d'en découvrir d'autres. Ce travail présente une partie du processus qui mène à ces nouvelles tendances.

Mots-clés : Data mining, Twitter, Text mining, Novelty detection, Nouvelles tendances, Social Network Analysis, Shampoing

Table des matières

Liste des tableaux

Liste des figures

Liste des équations

1. Introduction

A l'heure où le marché des données explose, le grand public s'interroge sur l'utilisation qui est faite de ces dernières. Dans quelle mesure des entreprises comme Facebook, Twitter ou Google, transforment-elles, en valeur marchande, les informations qu'elles récoltent sur nous ? Ce travail est un exemple d'exploitation des données du réseau social Twitter. L'analyse qui est réalisée, même si elle touche à des données personnelles, reste plutôt du bon côté de la morale. Le gain d'information qui est visé concerne les réactions d'une masse de consommateur et non celles d'un individu précis, ce qui est déjà plus discutable.

Peut-on découvrir de nouvelles tendances à partir des données de Twitter ? Une entreprise américaine avait mandaté l'université de Genève pour qu'elle tente de répondre à cette question. Après une restructuration, le projet a été mis de côté. Monsieur Kalousis, me sachant intéressé à effectuer mon mémoire de Bachelor dans son domaine, m'a proposé de le poursuivre en interne.

Le travail a donc pour but de découvrir les nouvelles préoccupations des utilisateurs autour d'un même sujet. Ici, il s'agit du shampoing et des cheveux. Quelles sont les dernières tendances qui occupent les utilisateurs de ce réseau social, quand ils parlent de leurs cheveux et de shampoing ? Nous allons tenter de répondre à cette question.

Pour découvrir ce qui est nouveau dans leur discours, il faut d'abord connaître ce qu'est leur discours en général. L'apprentissage de ce dernier passe donc par un regroupement des tweets en sujets majeurs. Ces sujets représentent une synthèse de ce qui intéresse le plus les utilisateurs.

Partant de cette connaissance de l'environnement, la phase de détection de nouveauté est simple. Chaque nouveau tweet va être confronté avec l'ancien jeu de données. Soit il est suffisamment proche d'un des sujets majeurs, alors il est considéré comme un tweet « déjà connu » ; soit le tweet est différent de tous les anciens sujets, alors il est considéré comme appartenant peut-être à un nouveau sujet. Si la liste de ces tweets « différents » est conséquente, un autre regroupement est tenté afin d'obtenir ces fameuses nouvelles tendances.

2. L'exploration de données

Le projet que j'ai eu à réaliser est un projet de « data mining » ou en français exploration de données. Cette sous-discipline de l'informatique a pour but d'optimiser le gain d'information sur différents types de données.

Le terme « data mining » est utilisé ici par abus de langage comme « knowledge discovery in databases (KDD) ». Ce dernier est un terme plus générique lorsque l'on parle du processus dans son ensemble, c'est-à-dire de la compréhension du besoin métier à la réalisation des outils techniques. Le terme « data mining » représente une des étapes du processus KDD, l'étape de la recherche du modèle le mieux adapté au problème rencontré et sa mise en application.

Il s'agit donc d'un processus qui demande des connaissances dans plusieurs domaines informatiques : la programmation et la base de données, ainsi que dans des domaines plus mathématiques comme les statistiques ou la géométrie (notamment pour les mesures de distance).

Les domaines d'application sont vastes : le marketing, la finance, la médecine, la biologie ou encore les services de renseignements. Le marketing est un des secteurs les plus importants de l'économie, c'est donc là que le « data mining » est le plus souvent utilisé. L'analyse que j'ai faite concorde avec cela, elle a pour but de mieux cerner les nouveaux intérêts des consommateurs.

2.1 Le processus KDD

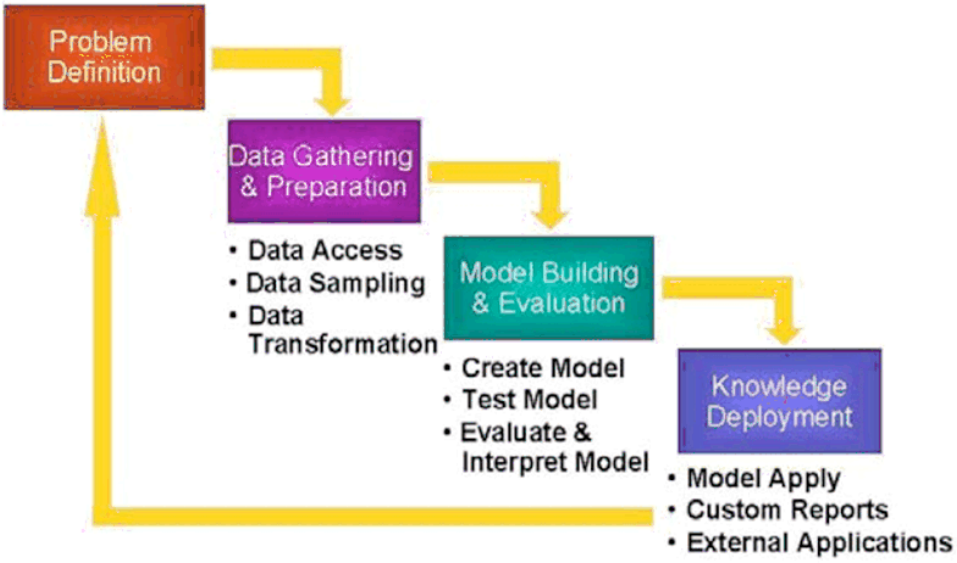
Le processus commence par la préparation des données. Cette étape est la plus longue, environ 80% (cf. cours de Monsieur Kalousis) de la démarche globale. Elle consiste à trouver la meilleure voie de récupération et à uniformiser les données pour ensuite pouvoir les traiter.

La seconde étape est la recherche d'un modèle applicable sur nos données, l'étape de « data mining ». Même si l'on a une idée du type d'information que l'on cherche, il existe souvent plusieurs modèles qui peuvent donner de bons résultats. Cette étape consiste à estimer quels algorithmes sont les plus adéquats en fonction de la structure des données (taille, disparité, nombre d'attributs, etc.) à traiter.

Une fois le modèle défini, il faut évaluer la qualité de nos résultats. Pour ce faire, il existe des méthodes statistiques qui peuvent déterminer le taux de fiabilité des

résultats obtenus. Une des plus connues est la méthode « ten-cross validation ». Néanmoins, comme ces méthodes ne s'appliquent pas au type d'analyse que j'ai réalisé, je ne m'étends pas sur ce sujet. Il est primordial d'indiquer quel degré de certitude l'on doit accorder aux résultats ; ces derniers doivent donc toujours être fournis avec cette réserve (dans la mesure du possible).

Figure 1 : KDD process



(source : oracle.com)

Les étapes décrites sont les étapes 2 « Data Gathering & Preparation » et 3 « Model Building & Evaluation », la fameuse phase de « data mining ».

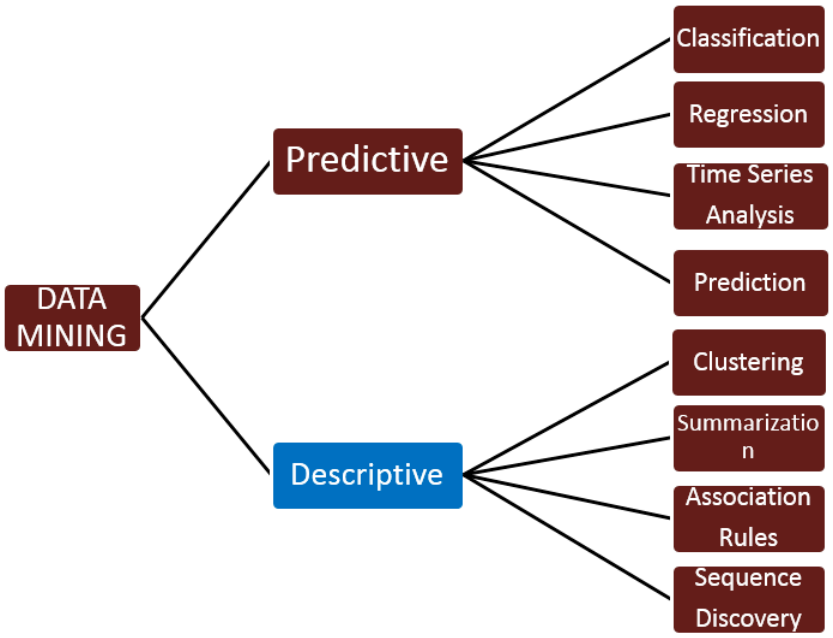
Elles constituent les parties techniques de la discipline. Ces étapes n'arrivent qu'après une analyse des besoins du corps métier ou la volonté d'une meilleure exploitation des données d'entreprise (étape 1 « Problem Definition »). La finalité de ce processus est un résultat qui doit être, le plus souvent, compréhensible par les décideurs de l'entreprise, c'est l'étape 4 « Knowledge Deployment ». Sur la figure, il est indiqué plusieurs types de résultats tangibles :

- l'application du modèle découvert et validé par l'étape de « data mining ». Il peut se présenter sous la forme d'un programme aidant à la prise de décision (finance, médecine, etc.).
- un rapport personnalisé en réponse à la problématique que l'on souhaite résoudre (étape 1). Le marketing souhaitant mieux cerner des segments de consommateurs ou des facteurs déterminants pour leur processus d'achat.

Pour ce dernier type d'analyse, il est ainsi important de garder en tête comme objectif une vulgarisation de ces informations complexes qui pourront entrer en compte dans les décisions managériales.

2.2 Les algorithmes de classification

Figure 2 : Les famille d'algorithmes de « data mining »



(source : techathon.com)

Pour comprendre et resituer dans son contexte le projet qui m'a été confié, il est important de visualiser les modèles (patterns) de « data mining » les plus courants.

Le cours de Monsieur Kalousis, qui m'a inspiré pour ce travail de Bachelor, traite essentiellement des algorithmes de classification. Ils font partie de la famille des algorithmes prédictifs (contrairement à ceux présentés dans ce travail). Ils sont appelés à découvrir le comportement probable (appelé résultat) des différentes instances pour lesquelles ce comportement n'est pas encore connu. Ils établissent une relation entre les attributs d'une instance et son résultat. Ce lien varie en complexité selon l'algorithme utilisé.

Un des algorithmes qui a le plus de succès dans la pratique fait partie de cette catégorie, il s'agit de « Support Vector Machines (SVM) ». Pour les problèmes linéaires, SVM crée un hyperplan qui sépare les labels des deux classes cibles en choissisant comme hyperplan celui qui crée la marge maximale. Une autre qualité

essentielle de SVM est qu'il ne se limite pas aux problèmes linéaires. Il peut (avec les paramètres appropriés) chercher à résoudre le problème en utilisant l'aire comme marge moyenne. Ces possiblités permettent à SVM de résoudre une gamme de problèmes variés.

La présente analyse fait partie des algorithmes descriptifs, la deuxième famille. Il s'agit d'une tentative de « clustering » sur des données. En bref, on essaie de regrouper des documents en n groupes les plus représentatifs. Elle sera expliquée plus en détail tout au long de cette analyse.

3. Le projet

Monsieur Kalousis et l'un de ses collègue de l'université de Genève étaient en contact avec une entreprise américaine qui souhaitait utiliser les réseaux sociaux pour mieux connaître ses clients. La première analyse qu'ils ont faite ne se limitait pas à Twitter (comme c'est le cas pour celle-ci) mais utilisait également des données venant d'Amazon (« Amazon customer reviews »).

Le projet initial a été abandonné, pour cause de restructuration de ladite entrerprise. Après quelques mois, Monsieur Kalousis m'a recontacté pour me dire qu'il était malgré tout intéressé à poursuivre le projet en interne, ce qui conduirait dans le futur à proposer ce type d'analyse à d'autres clients potentiels (des entrerprises suffisamment grande pour que la masse de données soit traitable).

La partie du projet qui m'a été confiée est donc la découverte de nouvelles tendances sur Twitter (« novelty or trend detection on Twitter data »). L'idée est de récupérer tous les tweets sur un sujet précis (à l'aide de mots-clés) et de comparer les tweets plus récents avec les anciens tweets qui auront été catégorisés. Si de nombreux tweets ne trouvent pas leur place dans les catégories plus anciennes (parce que trop différents), on essaye alors de les regrouper entre eux. Si de nouveaux groupes apparaissent (avec suffisamment d'instances), on peut considérer qu'un nouveau sujet de « préoccupation » est apparu. C'est là notre objectif.

La compagnie possède une des marques de shampoing les plus connues, c'est pourquoi toute l'analyse que j'ai réalisée est orientée vers ce sujet. L'analyse préliminaire de l'université de Genève a aussi été faite en traitant des données sur le shampoing et sur les marques phares de ce secteur.

3.1 Quelques analyses possibles

Ici je vais vous présenter quelques analyses possibles que l'on peut faire grâce aux réseaux sociaux. Ils viennent du projet qui a malheureusement avorté. J'ai pu avoir accès aux résultats préliminaires que l'équipe avait obtenus.

On parle d'analyse prélimiaire lorsque l'on réalise une maquette d'outils développables pour intéresser les clients. On peut ensuite approfondir avec eux ce qui leur paraît le plus pertinent.

3.1.1 Nouvelles tendances

Figure 3 : Trend detection

(source : projet Kalousis)

Le but de cette analyse est de découvrir l'évolution des occurrences des mots associés au terme de la recherche (sur la figure : « shampoo »). Si le mot s'affiche en grand et en vert, cela veut dire que ce dernier est apparu beaucoup plus dans les dernières 24h qu'auparavant. Au contraire, si le mot est petit et d'une couleur proche du brun, le mot est apparu moins souvent dans les dernières 24h.

On voit là l'intérêt pour la marque d'être mise au courant très vite si beaucoup de mots négatifs (comme ici « blind ») apparaissent récemment sur les disscussions autour du shampoing. Elle peut alors prendre les mesures qu'elle juge nécessaire pour tenter d'améliorer son image. L'objectif du « data mining » est de fournir une information compréhensible, c'est ensuite aux managers d'en tirer des conclusions.

3.2 Regroupement par sujets

Figure 4 : Automatic discovery of « topics » in a collection of documents

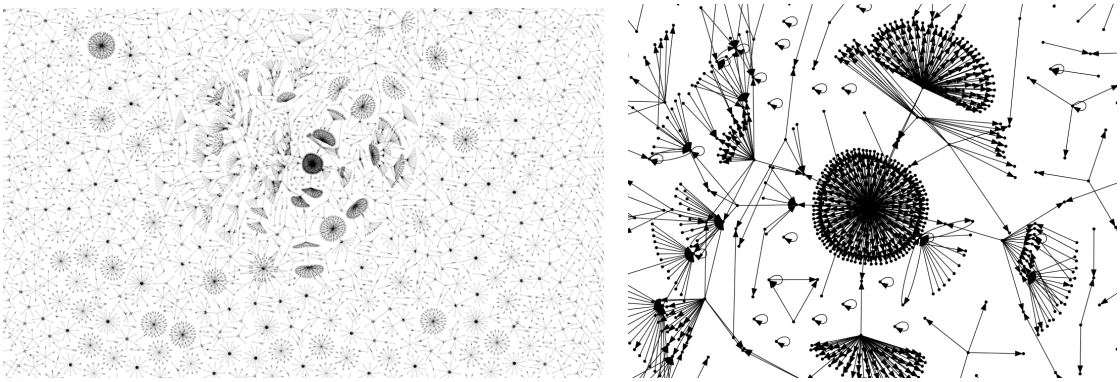
ɔic4	Topic5	Topic6	Topic7	T
ıing	hairs	entire	get	tr
m	tube	behind	cream	lil
n	conditioner	using	purchase	he
ghing	throughout	heavy	car	he
ılthy	done	getting	leaves	sa
e	thinning	bounce	shampoos	ef
	helped	towel	curl	ar
ıd	switched	place	lots	ve

(source : projet Kalousis)

La volonté est ici de regrouper les mots qui apparaissent souvent ensemble dans les documents (un document peut être un tweet ou encore un commentaire sur Amazon). Ces regroupement forment des pseudo-sujets de conversation qui peuvent être interprétés ensuite comme étant les préoccupations les plus courantes des utilisateurs. L'intérêt pour le business est de ne pas passer de nombreuses heures à lire des milliers de commentaires, ils peuvent se contenter de connaître ces regroupements succincts et gagner ainsi beaucoup de temps.

3.2.1 Découverte des acteurs les plus influents

Figure 5 : Automatic detection of most influential social actors



(source : projet Kalousis)

Voilà une autre analyse intéressante. Quels sont les acteurs les plus influents d'un marché ? Ici chaque trait correspond à un « retweet » (il sera expliqué dans le point

« 4. Twitter »). Le mot-clé de la recherche est ici « beauty », à gauche l'on voit l'ensemble des retweets et à droite, un zoom sur la zone centrale, la plus dense.

Sur la figure de droite, le gros amas peut être considéré comme l'acteur le plus influent sur Twitter concernant les produits de beauté. Il a un nombre important de retweets, signe qu'il est plus diffusé que les autres acteurs du marché. Tous les acteurs qui paraissent plutôt importants (avec un certain nombre de tweets), sont des cibles intéressantes pour le marketing ciblé. Là encore, le « dataminer » doit comprendre les enjeux de son analyse pour le corps business et non interpéter les résultats.

3.3 Les objectifs

La partie récupération des données ayant déjà été réalisée, je n'ai donc qu'à me connecter à un serveur, à lui envoyer des requêtes pour accéder à ces dernières. Le projet est plutôt complexe, c'est pourquoi je ne peux en réaliser qu'une partie. L'objectif que je dois atteindre est de suivre toutes les étapes KDD jusqu'à la détection des nouveautés. Comme certains choix dans la construction des modèles de « data mining » sont assez subjectifs, chaque étape doit être indépendante pour que d'autres algorithmes puissent être testés après mon départ de ce projet.

Je ne m'occupe pas non plus de fournir une interface pour le client. Le résultat de mon projet est un script « Makefile » (voir plus bas) qui crée tous les fichiers résultats dans un répertoire cible.

3.4 Déroulement du projet

Le projet est articulé en plusieurs phases distinctes. Certains choix ne sont pas optimaux en raison de la contrainte de temps pour un travail de Bachelor (environ deux mois de travail à plein temps).

La première partie est la phase de réflexion sur la problèmatique et sur l'environnement de travail. Comme il s'agit d'une sorte de mandat de l'université de Genève, j'ai simplement pris acte de la problèmatique posée. Monsieur Kalousis et Monsieur Nguyen m'ont expliqué ce qu'ils cherchaient à obtenir. Ils avaient à disposition une base de données contenant des milliers de tweets (environ 40'000 par mois), qui correspondaient à la liste exhaustive des tweets issus des termes de recherche « hairs » et « shampoo » (cheveux et shampoing).

La deuxième phase a été de récupérer les données et de les traiter. J'ai donc accédé au serveur et les ai enregistrées en fichier plat (CSV). Ensuite, il a fallu les uniformiser. Pour ce faire, j'ai appliqué les outils de « text mining » classiques pour réduire le texte à ses mots-clés. Il m'a encore fallu créer trois matrices pour obtenir de bons résultats avec les algorithmes de « clustering ». La première est une matrice d'occurrences, la deuxième une matrice de poids des occurrences pondérées par leur contexte et la dernière est une matrice de distances.

Possédant cette matrice de distance, j'ai pu créer mes regroupements sur la première période. Chaque élément principal de chaque regroupement a ensuite été comparé avec les tweets d'une deuxième période suivant la première. Les éléments en marge du regroupement ont été mis de côté puis on a tenté de les regrouper. Ce sont ces groupes qui constituent nos nouvelles tendances possibles.

La partie technique de ce travail est constiuée à chaque étape d'un script. En tout, 9 scripts produisent chacun un résultat qui sert le ou les suivants, jusqu'à nous amener aux nouvelles tendances. Ces scripts sont liés entre eux par le logiciel make, très utilisé sous unix. Il suffit de préciser les paramètres désirés (répertoire cible, dates des deux périodes de tweets et nombre de regroupements que l'on veut pour chacun des « clustering ») et le logiciel construit le résultat par la suite logique indiquée dans les points précédents. Quand le programme make se termine, il a créé 14 fichiers résultats.

4. Twitter

Comme les données que nous traitons proviennent de Twitter, il est important de connaître quelles sont les spécificités de cet environnement pour appliquer avec précision les outils durant le processus KDD.

Twitter est le deuxième réseau social le plus important après Facebook. Aujourd'hui, il compte plus de 500 millions d'utilisateurs dans le monde entier. Comme le plus souvent les utilisateurs ne rendent pas leur profil confidentiel, la plupart des données sont publiques (contrairement à Facebook). Comme les gens réagissent beaucoup sur l'actualité, il y a de nombreuses opportunités dans le traitement de ces données sur les phénomènes de masses (tendances, modes, comme c'est le cas dans la présente analyse).

4.1 Structure d'un tweet

Le message ne peut dépasser 140 caractères. Cette contrainte est très intéressante pour le « data mining », elle permet de gagner en clarté et de rendre les données plus facilement comparables.

Il survient en revanche un problème de cette contrainte de condensation, l'utilisateur va utiliser plus d'abréviations (plus difficilement déchiffrables) que sur les autres réseaux sociaux ou blogs dans lesquels il donne son avis.

Twitter est un réseau social international, il est donc possible que lorsque l'on récupère des tweets, ils ne soient pas tous dans la même langue. Comme souvent sur Internet, la majorité des documents (en l'occurrence des tweets) sont en anglais. Il y a donc un maximum d'opportunités d'analyse sur des sujets et des mots-clés en anglais. Les termes « hairs » et « shampoo » ne sont pas des mots couramment utilisés dans d'autres langues (comme c'est le cas pour de nombreux mots de la langue anglaise). Nous n'avons donc pas besoin de nous inquiéter du problème de la langue.

Dans leurs tweets, les utilisateurs réagissent beaucoup entre eux en se citant. Pour citer l'un de ses « following » (quelqu'un à qui l'on s'est abonné), il faut faire précéder le pseudonyme de l'utilisateur du signe « @ ».

Il est possible de répondre à un message publié par un autre utilisateur. Dans ce cas, l'utilisateur à qui l'on répond ainsi que les « hashtags » apparaissent au début de la réponse.

Comme nous l'avons vu dans les analyses faites par Monsieur Kalousis, il est possible de retweeter un message. Il apparaît ainsi sur le fil d'actualité de nos « followers » avec la mention « Retweeté par … ».

Les réseaux sociaux sont aussi très utilisés pour le partage de photos ou de liens. Sur Twitter, à cause de la contrainte des 140 caractères, la plupart des liens sont d'abord passés par les utilisateurs par des « URL shortening », ces sites créent une référence courte (stockée sur leur serveur) sur le lien de l'utilisateur. Ce dernier peut ensuite publier sur Twitter ce lien raccourci qui n'est qu'une redirection vers le vrai lien. Ainsi ce dernier occupe moins de caractères qu'une URL normale.

4.2 Les utilisateurs

Comprendre qui sont les utilisateurs de Twitter n'est pas chose facile. J'ai trouvé une seule étude sérieuse à ce sujet, elle vient du « think tank » américain « Pew Research Center ». Il a analysé 10 événements marquants de la politique américaine en 2012, en comparant les réactions sur Twitter et les sondages sur les mêmes sujets.

L'étude nous montre que pour la plupart des sujets, les réactions sont complétement différentes. En effet, les utilisateurs sont déjà très différents de la population ordinaire. Selon un sondage du New York Times, seuls 3% des adultes publient couramment sur Twitter et la moitié d'entre eux a moins de 30 ans. Comme nous montre le graphique suivant, les réactions sur Twitter sont très souvent en décalage avec l'opinion américaine.

Tableau 1 : Comparaison entre l'opinion publique et Twitter

When Twitter Reaction Was More Conservative than Public Opinion

Obama's second inaugural speech (Jan 2013)

Public opinion	%	*Twitter*	%
Positive	48	Positive	13
Negative	22	Negative	21
Neither/DK (Vol.)	29	Neutral	65

Opinion of John Kerry post-nomination (Dec 2012)

*Public opinion**	%	*Twitter*	%
Favorable	39	Positive	6
Unfavorable	36	Negative	32
No opinion/DK (Vol.)	26	Neutral	62

When Twitter Reaction Was More Liberal than Public Opinion

Obama's reelection (Nov 2012)

Public opinion	%	*Twitter*	%
Happy	52	Positive	77
Unhappy	45	Negative	23

Better job in the first presidential debate (Oct 2012)

Public opinion	%	*Twitter*	%
Obama	20	Obama support	59
Romney	66	Romney support	40

(source : pewresearchcenter.org)

Seuls deux sujets rejoignaient l'opinion publique américaine, il s'agit de l'approbation par la cour suprême de la conformité de la loi de l'assurance maladie obligatoire à la constitution ainsi que la nomination de Paul Ryan comme candidat à la vice-présidence.

Il est difficile de se faire un avis concret à partir de cette étude ; elle compare deux choses totalement différentes : des sondages où les gens sont choisis parce qu'ils sont représentatifs d'une population moyenne et un flux d'information où les gens réagissent non pas quand on les interroge, mais quand quelque chose les touche suffisamment pour réagir.

Dans notre analyse, l'étude a malgré tout un intérêt, elle nous montre que l'on ne pourra pas considérer a priori le résultat comme étant représentatif d'une population globale. Comme notre analyse concerne la détection de nouvelles préoccupations de consommateurs, on ne s'intéresse de toute façon pas de savoir ce que pense le consommateur moyen, mais simplement d'amener une information sur une alerte potentielle (nouveau sujet négatif) ou sur une tendance qui séduit de nombreuses personnes. Les résultats pourront être considérés a priori comme étant l'avis de personnes plus jeunes que la moyenne, très préoccupées par exemple, par leurs cheveux (au point d'en écrire un tweet).

5. Phase de « pipeline »

Cette phase, correspondant à 80% du travail en moyenne, d'après le cours de Monsieur Kalousis, est primordiale. Il s'agit de la récupération et du traitement préalable des données. Pour arriver dans de bonnes conditions à la phase d'analyse, il faut que les données soient ramenées à l'essentiel.

La première partie du « pipeline » se fait dans le langage Perl, il s'agit d'un langage « (…) *particulièrement adapté au traitement et à la manipulation de fichiers texte, notamment du fait de l'intégration des expressions régulières. » (Wikipédia). Comme il faut uniformiser les tweets avant de les traiter, ce langage est bien adapté.*

5.1 Récupération des données

Les données ont été récupérées depuis la base de données Twitter et stockées sur un serveur de la HEG, accesible on-line. Cette partie a été réalisée par des membres du corps enseignant de la fillière information documentaire (ID) de la HEG.

La première tâche a donc été de récupérer les données sur le serveur et de les écrire dans un fichier plat (fichier CSV en l'occurrence). Le langage utilisé est le langage Perl. Le serveur de la HEG qui contient les données est un « serveur Apache Solr ». Il permet de faire des requêtes Solr ; ces requêtes ont la particularité d'être inscrites directement dans L'URL.

http://localhost:8983/solr/get?id=1

Voilà à quoi ressemble une requête Solr ; ici on récupère les données qui ont un identifiant qui est égal à 1. Le retour du serveur peut être soit en JSON, soit en XML.

La requête que je fais sur le serveur de la heg est évidemment beaucoup plus complexe. Je dois entrer une date de début et une date de fin, afin d'avoir un intervalle ; il correspondra soit à mon « clustering » de départ (par exemple les deux premiers mois) et un deuxième intervalle qui correspondra aux données à partir desquelles on tentera de trouver de nouveaux sujets de préoccupation (en le comparant au premier intervalle).

Le serveur, par défaut, ne retourne que les 1000 premières instances. Monsieur Phong Nguyen désirait que l'analyse se fasse sur l'entièreté des données ; il m'a donc demandé de récupérer chaque fois l'ensemble des données d'un intervalle. Pour ce faire, il m'a suffi de faire une première requête sur l'intervalle où je précise 0 comme

nombre d'instance. Dans les meta données du XML, il est indiqué combien de données correspondent à ce champs de recherche. Je récupère ce nombre et relance une deuxième requête en indiquant le nombre total.

Voici le descriptif « pas à pas » du premier script de récupération.

1. vérification des arguments
2. interrogation du serveur pour connaître le nombre de résultats correspondant à la requête Solr
3. récupération du nombre total de documents correspondant à la requête
4. interrogation auprès du serveur de la même requête en lui demandant d'afficher le nombre de données qui correspondent à la requête (pt 3.)
5. récupération des données en XML
6. création d'un fichier de résultat CSV
7. parcours de chaque document XML
    7b. récupération du tweet, de l'auteur et de la date
    7c. sauvegarde des mots "hashtagés"
    7d. nettoyage du tweet, suppression des « hashtags », des URLs, des utilisateurs cités (@userx), de la ponctuation et des numéros
    7e. réinsertion des mots "hashtagés" dans le tweet « nettoyé »
    7f. inscription du tweet, de l'auteur et de la date, dans le fichier de sortie
8. fermeture du fichier de résultat
9. fin de la récupération

Ce petit descriptif permet à un lecteur peu familier du langage Perl de comprendre les actions majeures du script.

L'auteur et la date ne sont conservés ici que pour s'assurer de la cohérence des données.

Un point intéressant est le « 7d. ». Pour l'analyse de détection de nouveautés, les URLs (la plupart du temps sous une forme raccourcie) n'ont pas de pertinence. Les utilisateurs cités sont également supprimés puisque l'on ne s'intéresse pas ici aux liens qui existent entre utilisateurs. Les mots « hashtagés » sont conservés en l'état. On considère que si l'utilisateur a choisi de les souligner, ils ont une importance certaine pour lui et on ne pas pas le considérer comme un mot normal.

5.2 Préparation des données

Une fois les tweets qui nous intéressent récupérés, il faut uniformiser leurs contenus pour que l'algorithme d'analyse ne considère pas comme différents des mots proches. Il ne faut pas non plus qu'il perde en efficacté à cause d'un nombre trop important de mots connecteurs qui ne sont pas pertinents dans une analyse où l'on souhaite découvrir de nouveaux sujets de conversation.

Le descriptif de l'algorithme est le suivant :

1. validation du paramètre
2. ouverture du fichier CSV fourni par xmltocsv
3. récupération du dictionnaire du "Stemming" anglais dans le package Lingua
4. lecture du fichier CSV
    4b. sauvegarde des mots "hashtagés"
    4c. suppression des "Stopwords" contenus dans la liste de Lingua
    4d. utilisation du "stemming" sur les mots restants
    4e. suppression des mots qui ne possèdent qu'un caractère
    4f. réinscription des "hashtagés" dans le tweet
    4g. ajout dans un dictionnaire de l'occurence des mots, des mots de chaque tweet ; table associative clé (mot) - valeur (occurence), elle sera utile pour la suppression des mots à faible occurence
    4h. ajout du tweet nettoyé dans un tableau
5. fermeture de l'ancien CSV
6. création d'un nouveau CSV pour recevoir les tweets nettoyés
7. parcours du tableau
    7b. suppression des mots à faible occurence (grâce au dictionnaire 4g.)
    7c. suppression des espaces en trop
    7d. suppression du premier et dernier espace
    7e. suppression des tweets vides
    7f. inscription du tweet nettoyé dans le nouveau CSV
8. fermeture du nouveau CSV

Le point « 4c. » est intéressant : qu'est-ce que la suppression des « Stopwords » ? Les « Stopwords » sont des mots connecteurs ou de langage courant qui n'ont pas de pertinence dans un contexte comme le nôtre où l'on cherche à conserver uniquement l'essentiel du tweet. Par essentiel on entend les mots clés qui peuvent le résumer. Les mots supprimés sont par exemple : « who », « with », « is » ou « about » (« qui », « avec », « est » ou « à propos de »).

Les mots qui restent reçoivent un traitement de « stemming » ou en français : de racinisation. Cela veut dire qu'une fonction s'occupe de réduire un mot à son radical, le radical d'un mot étant ce qui reste après la suppression du préfixe et du suffixe. Par exemple, le mot « redemander », après un traitement de racinisation, deviendra « demand » (re-demand-er). Cela permettra ensuite que les mots légèrement différents dans la matrice d'occurrences, ne soient pas considérés comme différents.

Pour avoir des documents bien nettoyés, il faut encore supprimer les mots à faible occurrence ainsi que les tweets vides après ces traitements.

5.3 Statistiques sur les documents

Phong Nguyen m'a demandé de récupérer quelques statistiques sur les données traitées afin de se faire une idée sur la disparité des données. J'ai également fait un script Perl pour obtenir quelques informations intéressantes.

1. vérification des arguments
2. ouverture du fichier CSV
3. consultation d'une ligne
3b. saut de la première ligne (en-têtes des colonnes)
3c. incrémentation de 1 d'un tableau associatif où la clé est la longueur du tweet
3d. consultation d'un mot du document
3d2. incrémentation du compteur du nombre de mots
3d3. incrémentation du dictionnaire de mots à la position du mot en question
3e. incrémentation du nombre de documents
4. fermeture de l'ancien fichier csv
5. affichage du nombre de documents
6. affichage du nombre de mots totaux
7. affichage du nombre de mots différents
8. parcours du tableau contenant les tailles des documents
8b. affichage du nombre d'occurences pour chaque taille
9. parcours du dictionnaire des mots
9b. affichage du nombre de fois où un mot apparaît et combien de mots apparaissent ce nombre-là de fois

Voici ce que le script affiche comme informations :

Tout d'abord, il donne des informations globables sur les données.

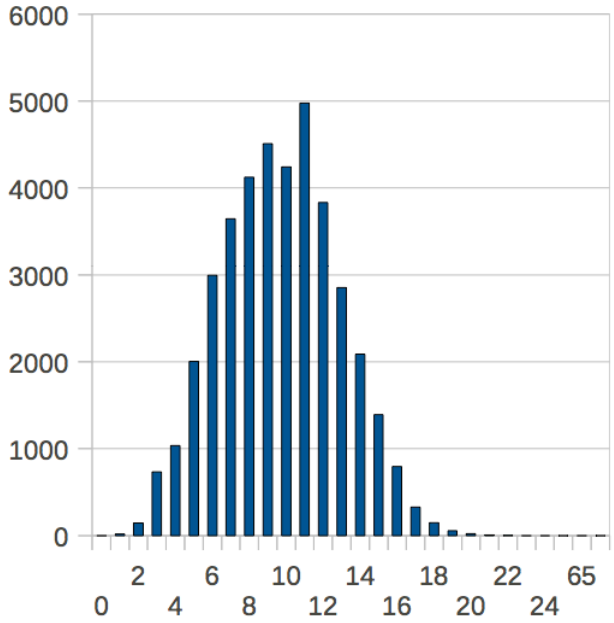
Nombre de documents : 39'975

Nombre de mots : 387'224

Nombre de mots différents : 5'599

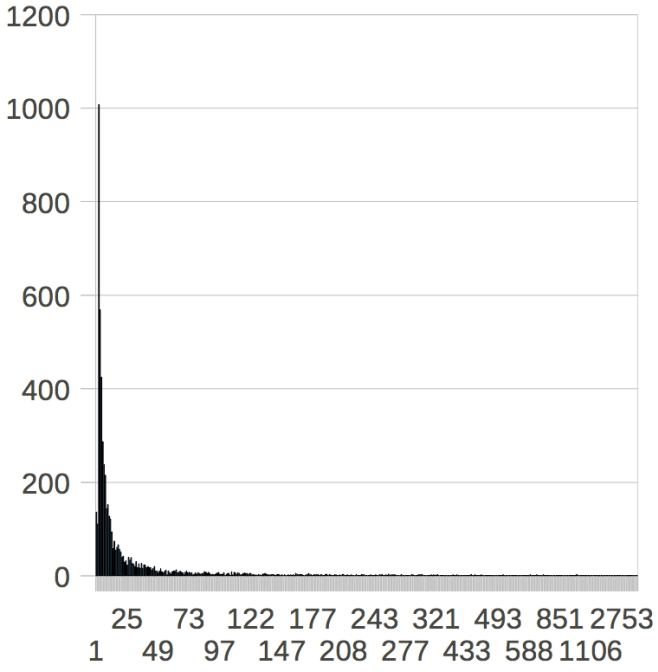
On voit là qu'il y a un nombre de mots différents très important (5'599). Un des problèmes qui pourrait survenir par la suite, c'est que la matrice ait du mal à faire des calculs avec autant de données.

Tableau 2 : Occurency of each document's size



Ce graphique représente le nombre d'occurrences (vertical) qui existe pour chaque taille de document (horizontal). On observe ici que les tailles des tweets (nombre de mots) respectent la loi normale. Le nombre de mots moyen tourne autour de 10. Le plus petit tweet compte 1 seul mot et le plus grand 20 mots.

Tableau 3 : Occurency of each word number's apparition



Ce tableau peut aider à décider combien de fois un mot doit apparaître au minimum pour ne pas être supprimé dans la phase de nettoyage (« 5.2 Préparation des données »). En effet, comme vous le verrez après, la matrice « tf-idf » donne un poids plus important aux mots rares (qui n'apparaissent pas souvent) et inversément. Il y a par exemple 1008 mots qui apparaissent 3 fois. Les tweets qui auront ce mot en commun seront immédiatement considérés comme extrêmement proches, mais comme 3 documents (sur une masse d'environ 40'000) ne suffisent pas pour former un regroupement à eux seuls, on peut s'interroger s'il faut ou non conserver ces faibles occurrences.

Une deuxième réflexion nous a beaucoup interrogés, faut-il ou non supprimer les termes de la recherche ? Ils apparaissent tellement de fois qu'ils ne sont même pas présents sur le tableau.

mot, occurrence
shampoo, 41182
hair, 43287

Comme ces mots sont présents dans presque tous les documents, est-il pertinent de les garder dans les documents ? Une de nos peurs était que ces mots créent un regroupement « poubelle », que tous les documents qui n'ont pas réussi à rejoindre un sujet rejoignent ce sujet « poubelle ». Comme il est dit précédemment, « tf-idf» permet de donner un poids très faible aux mots qui apparaissent de nombreuses fois. Dans la présente analyse ils ne seront donc pas enlevés, mais cette hypothèse n'est pas exlue.

5.4 Matrice d'occurrence

Pour la matrice d'occurrence, j'ai dans un premier temps tenté de la réaliser dans le langage R (langage particulièrement adapté aux traitements statistiques), mais les fonctions étaient un peu trop coûteuses en ressources pour être efficaces. J'ai donc opté pour la création d'une matrice d'occurrence en Perl.

Voici les étapes de l'algorithme.

1. validation du paramètre
2. ouverture du fichier CSV provenant de cleancsv
3. ouverture d'un nouveau fichier CSV qui stockera la matrice d'occurences
4. parcours de l'ancien fichier
4b. saut de la première ligne (ligne d'en-tête)
4c. parcours de chaque mot du tweet
4c2. incrémentation du compteur du nombre total de mots
4c3. incrémentation de la valeur stockée dans le tableau associatif à deux dimensions ; la première clé correspond

au numéro du tweet et la seconde au mot que l'on incrémente

4c4. le mot n'existe pas dans le dictionnaire

4c4b. ajout du mot au dictionnaire associatif et stockage de sa position

/ cette Ètape nous permettra de savoir dans quelle colonne on devra incrémenter le compteur des occurences pour chaque mot de chaque document */*

5. inscription dans le fichier de sortie de la ligne des en-têtes

6. parcours du tableau des documents

6b. récupération de tous les mots d'un document

6c. stockage dans un tableau associatif de l'occurence d'un mot, la clé utilisée est la position du mot dans le dictionnaire des mots

6d. complétion de la matrice d'occurences avec un 0 dans chaque colonne où le mot n'apparaît pas dans le document

6e. quand un mot apparaît dans le document, on met son nombre d'occurences au lieu du 0, dans la bonne col.

6f. enregistrement de la ligne dans le fichier CSV

7. fermeture des deux fichiers

Ce que fait cet algorithme, c'est simplement de faire correspondre chaque mot de chaque document dans la bonne colonne. Le fait de le coder en dur plutôt que d'utiliser une méthode du langage R nous a fait gagner un temps de calcul considérable.

Le résultat est un fichier CSV dans lequel on trouve à la première colonne tout le champ lexical des données séparé par des virgules. Ensuite, chaque ligne correspond à un document avec le nombre d'occurrences de chaque mot pour chacun des tweets, toujours séparés par des virgules.

Petit exemple simpliste avec deux documents : « shampoing beaux beaux cheveux » et « shampoing cheveux lisses ». Dans le fichier CSV de la matrice d'occurrences, ils apparaîtront sous la forme suivante :

shampoing, beaux, cheveux, lisses

1, 2, 1, 0

1, 0, 1, 1

Bien sûr, les vraies matrices sont beaucoup plus grandes ; si l'on reprend la matrice en exemple au point précédent, la matrice a une taille de 39'975 x 5'599.

5.5 Matrice « TF-IDF »

Pour cette partie, nous avons décidé qu'il était plus simple de passer dans le langage R pour éviter les erreurs.

Jusqu'à présent, la démarche était tout à fait standard pour une analyse de « text mining », c'est à dire que peu de choix importants ont été effectués. Dans cette partie, nous prenons une orientation, pondérer le poids de chaque mot dans chaque document par « tf-idf» puis associer cette pondération avec une mesure de similarité cosinus (point suivant) entre tous les documents. Ce choix est principalement dû à cette mesure de similarité cosinus et sera expliqué au point suivant.

« TF-IDF » (« Term Frequency – Inverse Document Frequency ») est une mesure de pondération qui permet de relativiser l'importance d'un mot dans son contexte. On entend par contexte d'un mot, son importance au sein du document dont il fait partie (tweet) et sa représentativité dans les données (l'intervalle choisi sur le serveur). En substance, cela signifie que plus un mot apparaît dans le document, plus le poids de ce mot augmente, mais si ce mot apparaît dans beaucoup d'autres documents, alors son poids diminue. Les mots rares sont donc privilégiés lorsqu'ensuite, on compare des documents entre eux.

Équation 1 : TF-IDF formula

$$\mathrm{tf_{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad \times \quad \mathrm{idf_i} = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|}$$

(source : academic.ru)

Pour calculer le « term frequency », on divise le nombre de fois où le mot apparaît dans le document par le nombre total de mots dans ce même document.

Le « inverse document frequency » s'obtient en calculant le logarithme du nombre total de documents divisé par le nombre de documents où le mot apparaît au moins une fois.

Le poids s'obtient en multipliant le « tf » et l' « idf ». Le résultat est donc une matrice de mêmes dimensions que la matrice d'occurrence où le nombre d'occurrences est remplacé par le poids « tf-idf ».

Figure 6 : Algorithme de TF-IDF

```
for(i in 1:rows){
  # Total of words in the doc
  wordsDoc <- sum(occurencyMatrix[i,])
  for(j in 1:columns){
    # When there is no instance of a word we already know that the tf-idf = 0
    if(occurencyMatrix[i,j]==0){
      newMatrix[i,j] <- 0
      next
    }
    newMatrix[i,j] <- occurencyMatrix[i,j]/wordsDoc * log10(rows/length(which(occurencyMatrix[,j]!=0)))
  }
}
```

Pour chaque document, on commence par calculer le nombre de mots contenu. Ensuite on parcourt les mots un à un, on inscrit directement 0 dans la case si l'occurrence est nulle. Si elle n'est pas nulle, on calcule le poids « tf-idf » du mot.

5.6 Matrice de similarité cosinus

Cette mesure de similarité s'accorde très bien avec un vecteur de pondération « tf-idf ». Le choix d'utiliser cette méthode cosinus pour calculer la distance entre les documents se justifie par la structure éparse de nos données. En effet, la distance cosinus ne prend en compte, dans ses comparaisons, que les dimensions non-nulles (Wikipédia). Je vous rappelle que dans notre exemple, notre document le plus long était de 20 mots pour près de 6'000 mots différents possibles ; l'écrasante majorité des dimensions est donc nulle.

Équation 2 : Cosine similarity formula

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

(source : wikipédia.org)

La formule de la similarité cosinus s'inspire, comme son nom l'indique, de la notion de cosinus en trigonométrie. Pour rappel, il s'agit du rapport entre le côté adjacent (de l'angle cible) d'un triangle rectangle par la longueur de son hypoténuse. C'est simplement le degré d'inclinaison de l'un des angles aigus du triangle.

Le résultat de la similarité est une mesure entre 0 et 1 où 1 signifie un angle de 0 degré, donc une parfaite similarité entre les deux documents comparés. 0 à l'inverse,

signfie un angle maximal de 90 degrés, soit aucune similarité entre les documents. Il existe encore le cas d'une opposition entre deux vecteurs où l'angle pourrait aller jusqu'à 180 degrés, mais dans notre cas, où nous comparons des mesures « tf-idf » calculées depuis des matrices d'occurrences, les documents sont au maximum sans similarité.

L'équation est donc : la somme de la multiplication des « tf-idf » des mots des deux tweets comparés (produit scalaire), divisée par la racine carrée de la somme de chaque mot (poids « tf-idf ») au carré pour le premier document, multipliée par le même calcul pour le second document.

Figure 7 : Algorithme de similarité cosinus

```
for(i in 1:size){
    cosineMatrix[i, i] <- 1
    j <- i+1
    while(j <= size){
        # The Cosine Similarity of two vectors (d1 and d2)
        c <- runDistance(tfidfMatrix[i,], tfidfMatrix[j,])
        cosineMatrix[i, j] <- c
        cosineMatrix[j, i] <- c
        j <- j+1
    }
}
```

```
runDistance <- function(docA, docB){
    dot <- sum(docA*docB)
    d1 <- sqrt(sum(docA^2))
    d2 <- sqrt(sum(docB^2))
    cos <- dot / (d1 * d2)
}
```

J'ai séparé volontairement en deux parties l'algorithme de calcul de la matrice de similarité pour l'éclaircir. L'algorithme de gauche s'occupe du remplissage de la matrice et celui de droite est responsable du calcul de la distance cosinus entre deux documents.

Une matrice de similarité est une matrice carrée ; dans chaque colonne, le document a un indice de comparaison entre 0 et 1 avec les autres documents et inversément (dans chaque ligne, le document a…). C'est pourquoi l'indice est le même dans la case [i, j] et la case [j, i] (ligne, colonne). Le tableau peut se lire dans les deux sens. La deuxième ligne de l'algorithme de gauche s'explique également par cela, un document comparé à lui-même a forcément une similarité maximale.

La fonction de droite, appelée à la 6ème ligne à gauche, correspond au calcul de la distance cosinus entre les deux documents comparés (voir formule ci-dessus).

6. Phase de « clustering »

Les données étant suffisamment traitées pour être exploitées, il faut maintenant envisager d'appliquer un algorithme de « data mining » pour créer une méthode de « clustering ».

Une méthode de « clustering » (regroupement en français), consiste à créer des groupes cohérents parmi nos données. Il existe pour cela de nombreux algorithmes qui ont des résultats plus ou moins bons selon le problème. Cette classification prend son sens dans notre projet puisque l'on recherche les dernières tendances exprimées par les utilisateurs. Pour trouver ce qu'il y a de nouveau, il faut commencer par connaître quelles étaient les anciennes préoccupations. Le « data mining » fournit une famille d'algorithmes qui permet la résolution de ce type de problème, ce sont ces méthodes de « clustering ».

Pour ce projet, nous avons d'abord choisi d'essayer un regroupement dit « hiérarchique » (voir point suivant) qui n'a pas donné les résultats escomptés. On ne peut cependant pas affirmer que cette méthode soit incompatible avec notre projet puisque la phase de « pipeline » n'était pas aussi fournie que maintenant lorsque les tests ont été effectués.

Phong Nguyen et Alexandros Kalousis ont décidé, après ce premier échec, d'approfondir la phase de « pipeline » et de changer de méthode de « clustering ». La méthode utilisée actuellement est la méthode « k-mean».

Phong Nguyen aurait souhaité également faire des essais avec l'algorithme « Latent Dirichlet Allocation (LDA) » ; malheureusement, le temps a manqué pour que cette voie puisse être testée.

6.1 Les différentes méthodes

Il existe deux familles principales d'algorithmes de « clustering » ; il s'agit des méthodes de regroupement hiérarchique ou des méthodes de partitionnement des données (Wikipédia).

La méthode hiérarchique est dite « top-down » (de haut en bas) ; elle part d'un ensemble où sont contenues toutes les instances, puis divise cet ensemble en plus petits groupements et ainsi de suite (souvent des divisions binaires). A l'arrivée, chaque instance est classée dans une branche finale de l'arbre. Quand on a un

nombre de données important, comme c'est le cas ici, on coupe l'arbre après un certain nombre de divisions, cela évite d'avoir des groupes trop spécifiques.

La méthode agglomérative, à l'inverse, considère dans un premier temps tous les éléments séparément. Elle crée ensuite des regroupements par similarités jusqu'à ce que toutes les instances soient assignées à un « cluster » (groupe). Il faut préciser à la méthode combien de groupes on souhaite créer pour nos données (le paramètre k).

6.2 La méthode k-means

La méthode que nous utilisons ici est une méthode qui découle des algorithmes agglomératifs. Il faut pour l'utiliser préciser le nombre de groupes cibles, il s'agit du paramètre k. L'algorithme choisit dans un premier temps aléatoirement un nombre k de documents qu'il considère comme les k éléments centraux potentiels. On assigne ensuite toutes les instances à l'un de ces « clusters ». Il faut ensuite pour chaque cluster calculer un nouveau document central appelé « centroid ». L'algorithme est relancé avec ces nouveaux k documents centraux. L'itération continue jusqu'à ce que les points centraux se stabilisent (deux fois consécutivement).

Pour que cette méthode soit efficace, il faut tester de nombreux k et choisir celui qui offre le meilleur « clustering ». Pour cela, on peut récupérer la mesure moyenne de la cohésion des groupements (moyenne des distances carrées). Une autre solution d'amélioration consiste à mieux choisir les premiers documents qui font office de « medoids » pour la première itération. On choisit dans cette phase de construction chaque medoid de départ en s'assurant qu'il est à une certaine distance des autres avant de lancer l'itération.

6.3 La fonction PAM

PAM ou « Partionning Around Medoids » est une fonction du package R « cluster ». C'est avec elle que nous créons les regroupements depuis nos données.

Une des raisons du choix de cette fonction pour le « clustering » est que c'est une des meilleures fonctions qui implémente la méthode k-means et qu'elle nous permet de lui donner directement une matrice de distances comme paramètre. En effet, ce qui est problématique avec beaucoup de ces méthodes, c'est qu'elles attendent souvent en paramètre une matrice de données et non une matrice de distance. La distance est calculée dans la fonction, alors elle laisse à l'utilisateur le choix de la métrique parmi

une liste qu'elle fournit. Malheureusement, notre mesure de distance, la similarité cosinus, n'était pas disponible dans la majorité des méthodes. C'est pourquoi on a calculé la matrice de similarité cosinus, on l'a ensuite transformée en matrice de dissimilarité (étant donné que l'on sait que l'on reste dans l'orthant positif, la transformation est 1 – matrice de similarité), puis on l'a fournie en paramètre à PAM. Il s'est ainsi servi de notre matrice directement pour chercher les différents « clusters ».

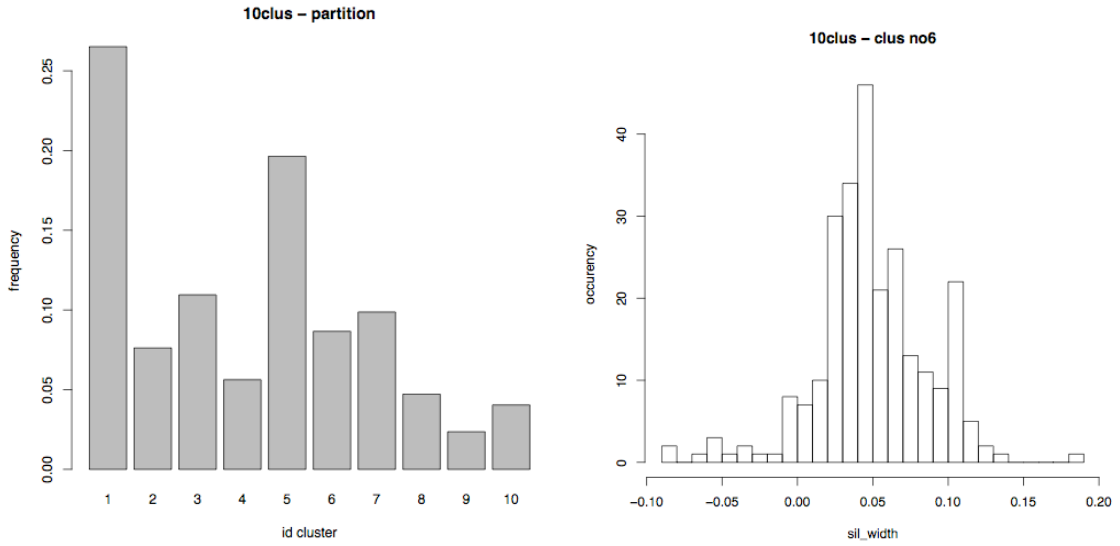
Les avantages de PAM sur « k-means » sont les suivants (MAECHLER, Martin. CRAN - Package 'cluster', 26-03-2013) :

- il accepte les matrices de dissimilarités
- il est plus robuste parce qu'il minimise la somme des dimissimilarités alors que « k-means » minimise la somme des distances euclidiennes carrées
- il fournit un affichage graphique intéressant : le « Silhouette plot »
- on a une mesure pour calculer la qualité des regroupements (« pam$silinfo$avg.width »)

L'objectif final de PAM est de trouver les k documents les plus représentatifs en minimisant la somme de leurs dissimilarités.

Comme expliqué au point précédent, il y a une première phase dans l'algorithme de PAM qui sert à trouver des medoids de qualité avant la première itération. On peut également lui fournir une liste de documents cibles qui seront utilisés directement comme medoids (il faut connaître les bons regroupements au préalable).

Figure 8 : Clustering partition & cluster 6

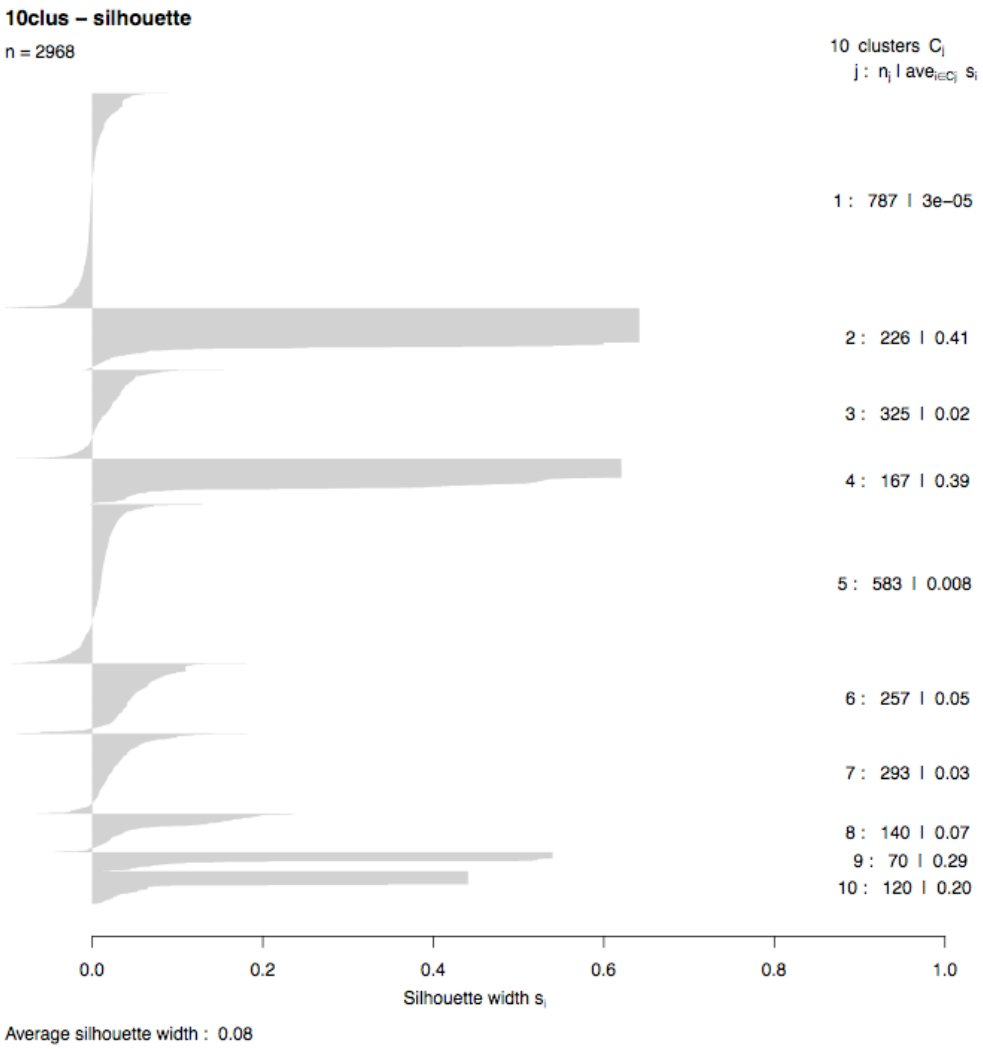


Voilà deux graphiques correspondant à un regroupement de PAM avec k = 10 sur 3000 données. A gauche, le graphique nous montre la répartition entre les 10 « clusters », une répartition avec trop de différences entre les « clusters » signifierait probablement un mauvais regroupement.

Le deuxième graphique montre comment les documents sont répartis dans le groupe numéro 6. On observe la loi normale autour du « medoid » qui doit se trouver autour d'un sil_width de 0.05. Cela signifie que le « cluster » est plutôt de bonne qualité, on a donc de grandes chances d'être ici en face d'un des sujets importants de nos données.

Le « sil_width » est un indice pour estimer la cohésion d'un « cluster ». Plus les données du « cluster » sont proches de 1, plus le regroupement est considéré comme bien séparé des autres regroupements. Une observation plus proche de 0 signifie qu'elle se trouve entre deux « clusters ». Si le « sil_width » de l'observation est négatif, elle n'est probablement pas dans le groupe.

Figure 9 : Silhouette plot



Le « Silhouette plot » est le nouveau graphique fourni par PAM. Il est très intéressant et vient en complément des deux graphiques précédents nous aider à évaluer la qualité de notre « clustering ».

De nombreuses informations sont présentes : tout en haut, n = 2968, est le nombre de documents observés. A droite, pour chaque « cluster », on a son numéro suivi du nombre d'éléments contenus et la moyenne du « sil_width » pour ce cluster. Le graphique central représente chaque observation des « clusters », en commencant par ceux qui ont le plus haut « sil_width » jusqu'à ceux qui en ont un négatif. Pour finir le « Average silhouette width », au bas du graphique, donne le « silhouette width » moyen pour l'ensemble du document.

Grâce à cette dernière information, on peut par exemple lancer 99 fois PAM avec un k de 2 à 100 et garder la moyenne des largeurs de silhouettes la plus haute pour notre « clustering ».

On pourrait penser que des « clusters » comme le 2 ou le 4 sont de très bons groupes parce qu'ils ont les meilleurs « sil_width » moyens. Cependant, comme ils ne respectent pas la loi normale, on peut s'interroger sur leur contenu. C'est un effet de bord causé par mon exemple qui n'est que de 3000 données ; il ne s'agit pas de sujets importants mais de nombreux retweets. Comme proportionellement à la masse, ils sont importants, ils ont créé un groupe pour eux (puisque la dissimilarité entre deux retweets est nulle). Ce problème a moins de chances de survenir avec plus de données, mais peut très bien réapparaître si la valeur de k est élevée.

6.4 Sélection du paramètre k

Sélectionner le paramètre « k » est une des choses les plus délicates pour le « clustering » et par conséquence pour la détection de nouveauté aussi. Comme nous faisons deux « clustering » ; un premier sur le jeu de données « ancien » et un deuxième sur les « outliers » pour découvrir les nouveaux centres d'intérêts, le deuxième va dépendre fortement du premier. On ne peut pas demander au premier un nombre trop faible de « clusters », sinon il serait impossible de découvrir des « outliers » car la dispersion de chacun de ces « clusters » serait énorme.

La première idée testée s'inspire du package R « fpc » ; il possède une fonction qui s'apelle pamk. Elle optimise la silhouette moyenne en testant différents « k » possibles.

Malheureusement, au vu du grand nombre de nos données, la fonction n'a pas donné de résultats. J'ai donc reproduit la fonction pamk, en écrivant un script qui teste les données avec différents « k » possibles. Cette fonction marche mais n'a pas de résultats aussi séduisants qu'on pourrait le penser. Au vu de nos données éparses, plus le « k » est grand, plus la « average silhouette » est grande. Je n'ai pas testé plus loin que 200, car déjà là, le temps de calculcs pour 4000 documents était considérable. D'autant plus que si par la suite, dans le rapport, on parle des 200 sujets essentiels autour du shampoing, l'information est trop floue pour améliorer la compréhension d'un manager. Cette optique de tâtonnement semble donc être une mauvaise idée, en tout cas en se basant sur la silhouette moyenne.

L'alternative qui a été retenue, sans être optimale, est de préselectionner un nombre « k » choisi sans base solide comme l'aurait été la silhouette moyenne. Après quelques tests, 15 ou 25 « k » semblent être de bonnes opportunités : 15 parce qu'il représente le bond le plus spéctaculaire (depuis la silhouette moyenne de 10, voir annexes 1 et 2), et 25 parce que c'est un nombre plus élevé, donc plus propice à la détection de nouvelles tendances et qu'il a aussi une bonne silhouette moyenne. Pour cette analyse prélimiaire, j'ai donc choisi ce dernier. Dans la même logique, j'ai tenté un « clustering » avec k = 5 pour les « outliers » (dont probablement 4 « vrais sujets » et 1 « sujet poubelle »). Cela représente environ 15% du nombre du premier regroupement ; c'est beaucoup, mais la disparité de nos données est grande ce qui va pousser de nombreux tweets à être rejetés par les regroupements initiaux.

7. Détection de nouveauté

Une fois les regroupements obtenus, l'étape suivante est la détection de nouvelles tendances. Cela consiste à comparer deux périodes de tweets : une première sur laquelle les regroupements ont été effectués et une seconde qui la suit (dans le temps). Les sujets principaux de la première période sont connus ; il s'agit des medoids récupérés avec la fonction PAM. Notre interrogation est donc la suivante : existe-t-il de nouvelles préoccupations qui émergent de la deuxième période ?

Pour répondre à cette question, chaque tweet de la nouvelle période va donc être comparé avec tous les medoids. Ce qui nous intéresse, c'est de savoir si un nouveau document est suffisamment proche du medoid avec lequel il a le plus de similarité, pour en faire partie. S'il est trop différent (relativement aux autres instances du « cluster »), il sera considéré comme un « outlier » (dans le contexte : élément qui se trouve en marge). Le regroupement de tous les « outliers » formera peut-être nos fameuses « nouvelles tendances ».

7.1 Comparaison entre les medoids et les nouveaux tweets

Pour connaître la distance entre deux éléments, la similarité cosinus avait été calculée, puis transformée en dissimilarité pour obtenir le « clustering ». Ainsi, pour comparer un medoid (tweet de la première période) et un nouveau tweet (tweet de la seconde période), il faut les mettre sur la même échelle. Comme le « clustering » a été fait sur la première période, l'échelle sélectionnée naturellement est celle de la première période. Pour la comparaison, on a besoin de deux vecteurs pondérés par le poids « tf-idf ». Nous avons décidé qu'il était préférable de calculer ce poids en se basant sur la même matrice d'occurences que celle qui a servi au « clustering ».

Uniformiser les données, pour une comparaison, consiste donc à créer une nouvelle matrice pour les données de la deuxième période, en se basant sur celle de la première période. Cette matrice aura le même nombre de colonnes que la première matrice d'occurrences, en y ajoutant, à la fin, un nombre « x » de colonnes. Ce nombre « x » correspond au maximum de mots inconnus contenus dans un des tweets de la seconde période. Il n'y a pas de nécessité à créer une colonne supplémentaire par mot inconnu ; les comparaisons ne sont faites qu'avec les medoids et évidemment, aucun d'eux ne contient l'un de ces mots.

Les « tf-idf » de la nouvelle période vont donc être calculés sur la base de l'ancienne matrice d'occurrences. On aurait pû préférer à ce choix le calcul des « tf-idf » des nouveaux tweets sur la base de la nouvelle matrice d'occurrences. Le renoncement de cette possibilité s'explique par la structure du calcul « tf-idf ». Le « term frequency » ne varie pas selon la matrice d'occurrences, c'est un rapport entre chaque mot et la totalité des mots contenus dans un document. En revanche, l' « inverse document frequency » relativise le poids d'un mot parmis les données. Mathématiquement, cela donne : le logarithme du nombre de documents divisé par le nombre de documents où le mot apparaît. Ce poids-là peut donc beaucoup varier si un mot apparaît de nombreuses fois dans la première période et non dans la deuxième (et inversément). Un exemple flagrant de dérive que le choix du calcul des « tf-idf » de la deuxième période sur la base de leur propre matrice représenterait, est une comparaison entre un medoid et un nouveau tweet qui lui serait identique. Leur similarité devrait être de 1, mais comme leurs « idf » seraient calculés à partir de données différentes, ils auraient une similarité inférieure. Là où le problème est plus grave, c'est pour les tweets éloignés des medoids ; ils risqueraient d'être considéré comme « outilers » pour une mauvaise raison.

7.2 Sélection des « outilers »

Classer un nouveau tweet comme « outlier » ou comme faisant partie des groupes existants demande une petite réfléxion et surtout quelques tests.

Chaque document va être comparé avec le medoid dont il est le plus proche. C'est à partir de cette comparaison qu'on décidera s'il peut être considéré comme inclus dans le « cluster » ou comme un « outlier ». La formule logique que nous avons utilisée pour cette décision est la suivante.

Équation 3 : Outliers selection formula

$$d(x, m_i) - \mu_{D_i} > c \times \sigma_{D_i}$$

Où « d(x, mi) » représente la dissimilarité entre l'instance « x » et son medoids le plus proche, ensuite la moyenne de « D_i » où « D_i » signifie la matrice des dissimilarités de son « cluster » le plus proche.

Dans la partie logique de droite, « c » est le paramètre variable. Grâce à lui, on module l'exigence de classification en le mulitpliant par l'écart-type du « cluster » concerné.

Ainsi, si le résultat de la formule logique est vrai, l'instance est sélectionnée pour faire partie des nouveaux sujets possibles.

La sélection du paramètre « c » dépend de deux choses : le nombre de documents considérés comme « outliers » et la silhouette moyenne du « clustering » qui sera fait de ces nouvelles instances (avec un « k » de 5 pour le « clustering »). J'ai donc testé un paramètre « c » allant de « 0.25 », qui rend l'expression logique trop facilement respectable, une majorité d'instances sont considérées comme des « outliers », jusqu'à « 2 », qui au contraire, rendait l'expression trop stricte et qui les classait tous comme appartenant à l'ancien jeu de données.

Les deux valeurs les plus intéressantes de « c » sont 0.75 et 1. La première nous permet d'avoir environ 20% des nouvelles données classées comme « outliers » et la deuxième valeur, « c = 1 », est plus exigeante. Elle classe seulement 5% des données comme « outliers » mais les résultats du « clustering » sont très bons. Les tests des différentes valeurs de « c » sont disponibles en annexes 3 et 4.

Après m'être intéressé au contenu des données, c'est celles du « c = 0.75 » qui semblent le mieux correspondre. Le problème avec « c = 1 » est qu'il y a une large prédominance pour les tweets contenant des mots qui n'existaient pas dans les anciennes données. Les tweets qui ont des mots différents de ceux des « clusters » mais qui existaient dans les anciennes données sont péjorés parce que ces mots-là n'ont pas un « idf » maximal. Nous ne cherchons pas les nouveaux mots qui apparaissent, mais bien les nouveaux concepts, contrairement à l'analyse de « trend detection » de Monsieur Kalousis. Comme nous tentons le regroupement des nouveaux avec « k = 5 » (sur 25 pour le « clustering » de départ), avoir 20% des données à classer correspond tout à fait à ce rapport et donnera une certaine légitimité aux nouveaux « clusters », s'ils sont bons.

8. Résultats préliminaires

Comme l'analyse n'est pas finie et que certains des choix présentés ont besoin d'être confirmés par une approche plus scientifique, les résultats affichés ici ne sont qu'un extrait de ce que l'on pourra obtenir quand le projet sera totalement terminé.

La période de temps que j'ai testée est courte, 5 jours pour créer le regroupement de départ et 2 jours pour la détection de nouveautés. Les tests avec des données plus importantes prennent un temps encore trop important ; je ne les présente donc pas.

La première période va du 10 janvier 2013 jusqu'au 15 janvier. Cette période compte 7'471 documents et 2'185 mots différents. Voici les sujets majeurs qui ont émergé : (la totalité des sujets est disponible en annexe 5)

- le shampoing rend les cheveux secs
- le shampoing prend soin de mes cheveux
- le shampoing Pantene
- le nouvel après-shampoing rend les cheveux doux
- le shampoing stimulant pour la croissance des cheveux Revita (des laboratoires DS)
- les shampoings contre la chute des cheveux
- les chansons sous la douche pendant qu'on se savonne
- l'après-shampoing pour les hommes qui se teindent les cheveux
- le shampoing qui fait sentir les cheveux
- le shampoing Horse aide à la croissance des cheveux
- un bon shampoing qui fait des cheveux naturels
- les cheveux peuvent-ils ressembler à ce qu'on voit dans les publicités
- une prodigieuse utilisation du shampoing
- en avoir marre de mettre du shampoing
- le shampoing Silver rend les cheveux beaux et blonds étincelants

Ces sujets-là constituent nos regroupements de bases. Ce qui nous intéresse maintenant est de savoir si dans les jours qui ont suivi cette période, de nouveaux sujets ont émergé.

La deuxième période suit la précédente jusqu'au 18 janvier 2013.

Voici les 5 sujets qui découlent de la détection de nouveautés.

- gagne une gamme de shampoings et de l'après-shampoing Clear Scalp & Beauty Hair
- un druide peut même trouver la formule qui fait du bon shampoing
- il faut en prendre soin tous les jours et mettre de l'après shampoing
- sans signification
- gagne un échantillon gratuit de Pantene expert en étant membre de Vocal Point

On voit ici 5 sujets pas très intéressants. Aucune nouvelle tendance à signaler. Cependant, les sujets 1 et 5 nous rapportent probablement des actions marketing qui ont été relayées suffisamment pour faire parler d'elles.

Les entreprises qui possèdent Clear Scalp et Pantene auraient pu constater un certain succès des actions qu'elles ont menées ou simplement s'intéresser au déclencheur de l'engouement pour leurs marques.

En fouillant un peu les données, il est facile de retrouver tous les documents qui servent de medoids. Voici par exemple le tweet qui représente le premier sujet :

Figure 10 : Tweet medoid

(source : twitter.com)

Il peut être intéressant pour Scalp Beauty d'être au courant de la réussite ou non de cette campagne qu'elle a peut-être orchestrée avec le site de « bons plans » Nicole's Nickels. Pour les autres, l'intérêt serait de savoir quel est le meilleur moyen pour faire parler de leurs produits ; ici il serait éventuellement bon de contacter Nicole's nickels si grâce à eux on peut avoir une visibilité importante.

Voilà un exemple de gain pour une entreprise d'utiliser cette détection de nouveautés, lorsqu'elle sera complétement opérationnelle.

9. Conclusion

Le sujet qui m'a été proposé, la détection de nouvelles tendances sur Twitter, m'a passionné. Je suis un fervent utilisateur des réseaux sociaux et pratiquer l'une des voies d'utilisation de ces données-là m'a permis de mieux cerner l'intérêt que le marché leurs porte de plus en plus. Partir d'une réalité plutôt diffuse (une masse de tweets) et effectuer la démarche qui mène à dégager de cette réalité complexe, des informations porteuses de sens (des sujets majeurs ainsi que de nouvelles préoccupations), est intelectuellement très stimulant.

Si le « data mining » m'intéresse particulièrement, c'est que les tâches du praticien sont très diverses. Suivre une partie du processus KDD, pour la première fois, a été très instructif. Dans le cours de Monsieur Kalousis, nous nous étions contentés d'appliquer les algorithmes et d'analyser les résultats. Ce projet était plus large, il m'a donné une vision globale de ce processus. Les données de départ étaient brutes, le résultat attendu était une abstraction de ces données en informations tangibles ; j'ai pu observer l'évolution de ces données et de l'information dont elles étaient porteuses, tout au long du projet.

Pour provoquer cette transformation, il a fallu appliquer de nombreux outils. D'abord, un nouveau language de programmation pour moi : le language Perl. Bien que ce dernier soit plutôt austère, j'ai eu du plaisir à créer de nombreuses fonctions dont j'avais besoin. Dans les languages traditionnels, beaucoup de fonctions pratiques existent déjà ; avec Perl, beaucoup de choses doivent être faites « à la main ». La confiance que l'on porte ensuite dans nos scripts est supérieure ; on sait que ce qu'on a développé est en bonne adéquation avec notre contexte. L'autre language avec lequel j'ai travaillé est R. J'avais appris ce language pendant le cours de Monsieur Kalousis, je partais ainsi avec quelques bases. Cela m'a permis de progresser plus vite et aujourd'hui c'est un language que je maîtrise plutôt bien. Comme il est très bien adapté aux analyses de « data mining » par ses nombreuses fonctions optimisées pour les matrices et les calculs statistiques ; il est très agréable de programmer en R sur ce genre de projets.

Je souhaitais découvrir plus en profondeur le domaine du « data mining » et c'est chose faite. Avoir l'opportunité de participer à tout un processus m'a fait gagner en expérience ; j'ai pu constater dans mon approche certaines erreurs qui m'ont servies de leçon pour mes prochaines analyses. La principale erreur est de m'être un peu empressé à commencer la partie pratique ; j'aurais gagné du temps si j'avais mieux

contextualiser le sujet de la recherche (en l'occurrence « Twitter »), avant de commencer à coder. En effet, mieux cerner l'environnement permet d'appliquer les outils avec plus de précision, de mieux savoir ce que l'on cherche et ce que l'on peut y trouver. Le plaisir que j'ai eu en réalisant cette analyse me pousse à en faire d'autres, si l'occasion se représente. Mon orientation vers le « data mining » est aujourd'hui renforcée.

Glossaire

KDD : « knowledge discovery in databases » est l'ensemble du processus d'exploration des données qui commence par la compréhension du problème ou par l'identification que pourrait représenter des données. Il organise ensuite le traitement des données, l'analyse et l'application des modèles de « data mining », puis offre un résultat tangible au client sous forme de graphiques ou d'applications. (voir Figure 1.)

Data mining : l'exploration des données, en français, est l'une des phases du processus KDD. Elle consiste à trouver quel modèle est le mieux adapté au problème et à la structure des données et à l'application et la validation de ce modèle. Par abus de langage, on utilise parfois ce terme pour l'ensemble de l'analyse (au lieu de KDD).

Twitter : il s'agit du réseau social le plus célèbre après Facebook. Il compte plus de 500 millions d'utilisateurs. La plupart des messages postés par les utilisateurs (appelés tweets) sont publics. La contrainte est qu'un message ne peut pas dépasser 140 caractères, cela condense l'information. Sur Twitter, on a des « followers » et des « following » ; n'importe qui peut nous suivre et vice-versa.

Tweet : un message publié sur Twitter est appelé un tweet. Voir ci-dessus pour plus d'informations.

Think tank : institut indépendant à but non lucratif qui fournit des informations sur différents sujets en lien avec la politique. Il peut être vu comme un intermédiaire entre le monde académique et le journalisme politique de par sa vocation à apporter une information de qualité supérieure.

Solr (Apache Solr) : un serveur Apache Solr est une plateforme de recherche logicielle sous licence libre. La communication se fait par un protocole HTTP et founit les données en format XML ou JSON. (Wikipédia)

Json : « JavaScript Object Notation » est un format léger d'échange de données. (Json.org)

XML : « Extensible Markup Language » est un language de balisage qui permet de faciliter l'échange de données complexes. Json est un peu plus léger (donc plus rapide) que XML, on le préfera si l'information n'est pas trop complexe. (Wikipédia)

Perl : il s'agit d'un langage de programmation « (…) *particulièrement adapté au traitement et à la manipulation de fichiers texte, notamment du fait de l'intégration des expressions régulières. » (Wikipédia).*

R : est un logiciel libre et un language de programmation. Il est particulièrement bien adapté aux calculs statistiques et à la création de modèles de « data mining » grâces aux nombreuses techniques d'affichage graphique.

Bibliographie

Supports de cours

KALOUSIS, Alexandros. Introduction to Data Mining and Knowledge Discovery. In : *Data mining*. Haute Ecole de Gestion de Genève. 2011.

KALOUSIS, Alexandros. Support Vector Machines. In : *Data mining*. Haute Ecole de Gestion de Genève. 2012.

KUHNE, Michel. Recherche opérationnelle. In : *Concepts avancés et technologies actuelles*. Haute Ecole de Gestion de Genève. 2012.

Chapitres de livres

BIFET, Albert, HOLMES, Geoff, KIRKBY, Richard, PFAHRINGER, Bernhard. Twitter Evolving data streams. In : *DATA STREAM MINING : A Practical Approach*. Waikato : COSI, 2011. P. 110

BIFET, Albert, HOLMES, Geoff, KIRKBY, Richard, PFAHRINGER, Bernhard. Twitter Stream Mining. In : *DATA STREAM MINING : A Practical Approach*. Waikato : COSI, 2011. P. 157

GAMA, João. Change Detection. In : *Knowledge Discovery from Data Streams*. Boca Raton : Taylor and Francis Group, 2010. P. 33

GAMA, João. Clustering from Data Streams. In : *Knowledge Discovery from Data Streams*. Boca Raton : Taylor and Francis Group, 2010. P. 79

GAMA, João. Novelty Detection in Data Streams. In : *Knowledge Discovery from Data Streams*. Boca Raton : Taylor & Francis, 2010. P. 133

WITTEN, Ian H, FRANK, Eibe, HALL, Mark A. Output : Knowledge Representation - Clusters. In : *DATA MINING : Practical Machine Learning Tools and Techniques*. Burlington : Elsevier, 2011. P. 81

WITTEN, Ian H, FRANK, Eibe, HALL, Mark A. Algorithms : The Basic Methods - Clustering. In : *DATA MINING : Practical Machine Learning Tools and Techniques*. Burlington : Elsevier, 2011. P. 138

WITTEN, Ian H, FRANK, Eibe, HALL, Mark A. Implementations : Real Machine Learning Schemes - Clustering. In : *DATA MINING : Practical Machine Learning Tools and Techniques*. Burlington : Elsevier, 2011. P. 273

WITTEN, Ian H, FRANK, Eibe, HALL, Mark A. Moving on : Applications and Beyond - Text Mining. In : *DATA MINING : Practical Machine Learning Tools and Techniques*. Burlington : Elsevier, 2011. P. 386

WITTEN, Ian H, FRANK, Eibe, HALL, Mark A. The Explorer - Clustering Algorithms. In : *DATA MINING : Practical Machine Learning Tools and Techniques*. Burlington : Elsevier, 2011. P. 480

ZAHO, Yanchang. Clustering. In : *R and Data Mining : Exemples and Case Studies*. Chatswood : Elsevier, 2013. P. 49

ZAHO, Yanchang. Outlier Detection. In : *R and Data Mining : Exemples and Case Studies*. Chatswood : Elsevier, 2013. P. 59

ZAHO, Yanchang. Text Mining. In : *R and Data Mining : Exemples and Case Studies*. Chatswood : Elsevier, 2013. P. 97

ZAHO, Yanchang. Social Network Analysis. In : *R and Data Mining : Exemples and Case Studies*. Chatswood : Elsevier, 2013. P. 111

Pages web

HONG, Lichan, CONVERTINO, Gregorio, CHI, Ed H. Language Matters in Twitter : A Large Scale Study. In : *Le site aaai.org* [en ligne]. Mis en ligne le 5 juillet 2011. https://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/view/2856/3250 (consulté le 11 avril 2013)

MAECHLER, Martin. Package 'cluster'. In : *Le site CRAN.R-Project.org* [en ligne]. Mis en ligne le 26 mars 2013. http://cran.r-project.org/web/packages/cluster/cluster.pdf (consulté le 12 mai 2013)

MITCHELL, Amy, HITLIN, Paul. Twitter Reaction to Events Often at Odds with Overall Public Opinion. In : *Le site PewResearchCenter.org* [en ligne]. Mis en ligne le 4 mars 2013. http://www.pewresearch.org/2013/03/04/twitter-reaction-to-events-often-at-odds-with-overall-public-opinion/ (consulté le 12 juillet 2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 7 juillet 2013 à 18:23. http://fr.wikipedia.org/wiki/Perl_(langage) (consulté le 15.05.2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 26 juin 2013 à 11:26. http://fr.wikipedia.org/wiki/Xml (consulté le 28.06.2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 30 juin 2013 à 15:37. http://fr.wikipedia.org/wiki/Think_tank (consulté le 03.07.2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 6 juillet 2013 à 15:31. http://en.wikipedia.org/wiki/Cluster_analysis (consulté le 10.07.2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 7 avril 2013 à 22:52. http://fr.wikipedia.org/wiki/TF-IDF (consulté le 05.05.2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 29 juin 2013 à 13:51. http://en.wikipedia.org/wiki/Stemming (consulté le 30.04.2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 12 juillet 2013 à 20:45. http://fr.wikipedia.org/wiki/JavaScript_Object_Notation (consulté le 30.04.2013)

Réseau social. In : *Wikipédia* [en ligne]. Dernière modification de cette page le 11 juillet 2013 à 08:58. http://en.wikipedia.org/wiki/Cosine_similarity (consulté le 03.05.2013)

Annexes

Annexe 1 : Optimize « k » test 1

Rscript optimizek.R ~/mywork/results/cos_2012-12-01_2012-12-03.rds ~/mywork/results/tw2012-12-01_2012-12-03_stem.csv
Loading required package: methods
[1] "**********"
[1] "Number of medoids: 5"
[1] "avg.width"
[1] 0.01325546
[1] "**********"
[1] "Number of medoids: 10"
[1] "avg.width"
[1] 0.02878757
[1] "**********"
[1] "Number of medoids: 15"
[1] "avg.width"
[1] 0.04021646
[1] "**********"
[1] "Number of medoids: 20"
[1] "avg.width"
[1] 0.04299418
[1] "**********"
[1] "Number of medoids: 25"
[1] "avg.width"
[1] 0.05286506
[1] "**********"
[1] "Number of medoids: 35"
[1] "avg.width"
[1] 0.06650172
[1] "**********"
[1] "Number of medoids: 50"
[1] "avg.width"
[1] 0.08151291

Annexe 2 : Optimize « k » test 2

Rscript mywork/optimizek.R mywork/results2/cos_2013-01-10_2013-01-15.rds mywork/results2/tw2013-01-10_2013-01-15_stem.csv 25
Loading required package: methods
[1] "**********"
[1] "Number of medoids: 5"
[1] "avg.width"
[1] 0.01191449
[1] "**********"
[1] "Number of medoids: 10"
[1] "avg.width"
[1] 0.02092
[1] "**********"
[1] "Number of medoids: 15"
[1] "avg.width"
[1] 0.0301031
[1] "**********"
[1] "Number of medoids: 20"
[1] "avg.width"
[1] 0.03834507
[1] "**********"
[1] "Number of medoids: 25"
[1] "avg.width"
[1] 0.04750406
[1] "**********"
[1] "Number of medoids: 35"
[1] "avg.width"
[1] 0.05990438
[1] "**********"
[1] "Number of medoids: 50"
[1] "avg.width"
[1] 0.07176891

Annexe 3 : « C » parameter test 1

Old data : 2012-12-01 to 2012-12-03
New data : 2012-12-04 to 2012-12-05
Old number of medoids : 25
Outliers number of medoids : 5

C	out	in	out.avg.width
2	0	1430	0.0
1.5	0	1430	0.0
1	42	1388	0.03034314
0.75	206	1227	0.03261065
0.5	463	967	0.01354574
0.25	876	554	0.01138564

Annexe 4 : « C » parameter test 2

Old data : 2013-01-10 to 2013-01-15
New data : 2013-01-16 to 2013-01-18
Old number of medoids : 25
Outliers number of medoids : 5

c	out	in	out.avg.width
2	0	2838	0.0
1.5	7	2831	0.01084231
1	174	2664	0.07634909
0.75	551	2287	0.06970122
0.5	990	1848	0.05245571
0.25	1517	1321	0.04497475

Annexe 5 : Old clustering

nb	idMed"	twMed	dateMed	sdMed	avgSim
1	7290	shampoo hair	15.1.2013	0.23	0.06
2	21	dri hair shampoo	11.1.2013	0.10	0.21
3	69	hair care shampoo	13.1.2013	0.14	0.29
4	7404	panten shampoo	13.1.2013	0.17	0.37
5	6518	hair smell like shampoo	14.1.2013	0.15	0.25
6	5634	new shampoo condition make hair soft	14.1.2013	0.17	0.20
7	694	ds laboratori revita hair growth stimul shampoo ml bottl thin hair treatment	10.1.2013	0.25	0.24
8	34	hair loss shampoo	11.1.2013	0.14	0.31
9	6245	shampoo use wash hair	14.1.2013	0.11	0.17
10	4616	liam sing weird song shower like shampoo hair soap bodi zayn malik lt	13.1.2013	0.36	0.56
11	7178	shampoo better go first clean hair condition better leav hair silki smooth	15.1.2013	0.26	0.27
12	344	just men shampoo condition oz hair color	12.1.2013	0.12	0.19
13	229	shampoo make hair smell gt gt gt gt gt gt	12.1.2013	0.23	0.72
14	5206	hors shampoo make hair grow	13.1.2013	0.17	0.32
15	5138	get shampoo hair	13.1.2013	0.11	0.24
16	6370	good shampoo amp amp condition natur hair	14.1.2013	0.10	0.20
17	1167	can hair look like hair shampoo commerci	10.1.2013	0.13	0.21
18	6608	got shampoo hair	14.1.2013	0.13	0.31
19	6958	hair perfect wonder shampoo use lol	15.1.2013	0.26	0.30
20	5735	love shampoo hair smell good	14.1.2013	0.18	0.28
21	6465	shampoo hair shower	14.1.2013	0.09	0.23
22	2644	fuck put shampoo hair	11.1.2013	0.12	0.23
23	4613	woke hair bright yellow now sit silver shampoo shower cap utiful	13.1.2013	0.44	0.49
24	7243	need shampoo hair	15.1.2013	0.18	0.33
25	1546	shampoo condit hair	11.1.2013	0.13	0.32

Annexe 6 : Outliers clustering

idMed	twMed	dateMed	sdMed	avgSim	avgSim
1	439	enter win clear hair scalp beauti therapi shampoo condition	17.1.2013	0.40	0.24
2	328	potion master amp can even find right formula make great shampoo hair	16.1.2013	0.26	0.18
3	32	thank gt just take care hair time use shampoo condition everi day	15.1.2013	0.10	0.08
4	317	lol liam suka nyanyi shampoo hair soap bodi kalo lagi mandi	16.1.2013	0.30	0.22
5	531	free sampl panten expert collect agedefi shampoo amp condition vocalpoint member	16.1.2013	0.32	0.37

Contacts

Etudiant : Alexis Beck

afy.beck@gmail.com

Directeur de mémoire : Alexandros Kalousis

alexandros.kalousis@hesge.ch

Coordinateur du projet : Phong Nguyen

phong.nguyen@unige.ch