8INF954 – Rapport projet Automne 2017

Text Mining

Forage de données

Paul Michaud ; Alexandre Marteaux
01/12/2017

Table des matières

Introduction

Le Text Mining est une branche du Data Mining spécialisée dans le traitement de texte pour en analyser le contenu puis en extraire des connaissances. Les principales tâches à accomplir consistent en la reconnaissance de l'information présente dans le document et son interprétation. Tout cela est possible grâce à une recherche sémantique reposant sur l'analyse du langage naturel et la gestion de bases de connaissances spécialisées. On peut par exemple distinguer une plainte d'un client d'une demande d'information, ou même un spam d'un message publicitaire, en inspectant la tournure des phrases.

Avec l'avènement des réseaux sociaux, les sources et les contenus de données n'ont cessé de croître exponentiellement. Face à cette tendance, les entreprises maintiennent une volonté de conserver un maximum d'informations exploitables. La majorité de ces informations reste encore aujourd'hui inexploitée.

Les applications du text mining sont multiples, par exemple dans le marketing, il permet de parfaire sa connaissance du client, en analysant les enquêtes de satisfaction, les lettres de réclamation, les échanges téléphoniques, etc.
Dans le domaine des banques et assurances le text mining permet de prévoir le nombre d'accidentés sur l'année suivante, d'élaborer des profils précis ou encore de détecter les factures frauduleuses.
Enfin dans le domaine du biomédical, il permet d'analyser les publications liées à certains domaines du fait du nombre croissant de publications électronique.

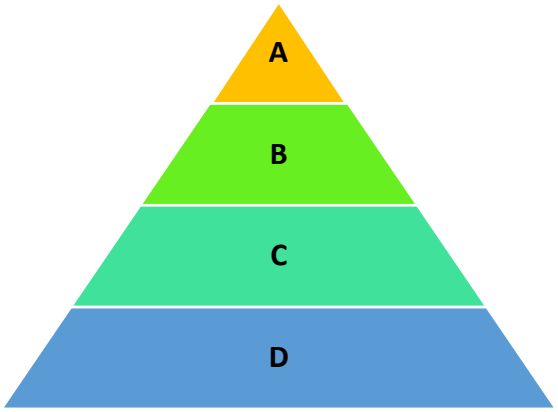
Le volume de ces données ne cesse de croître, on estime que 80% des données stockées par les entreprises sont actuellement inexploitées. Le text mining va permettre de ressortir les informations pertinentes de tous ces textes.

Technique d'analyse de texte

NLP natural language processing

NLP est un composant du text-mining qui performe une analyse linguistique qui permet à une machine de comprendre un texte en langage naturel.

Le NLP dispose de différent niveau d'interprétation du langage :



- **A** : Pragmatique
  - Prise en compte du **contexte.**
- **B** : Sémantique
  - **Utilise le sens** du mot issu d'un dictionnaire.
- **C** : Syntaxique
  - **Analyse la relation** des mots dans une phrase.
- **D** : Lexical
  - **Segmentation** du document en phrase et mots.

Pre-processing

NLP inclut plusieurs méthodologies pour distinguer les ambiguïtés dans le langage naturel :

- **Séparation** des phrases dans un text
- **Normalisation** des mots qui ont plusieurs orthographes. Par exemple, anglais américain et britannique labeled/ labelled, extra-terrestrial / extraterrestrial
- "**Stemming"** qui permet de considérer les mots qui sont de la même famille (computer / computation).
- **Morphological analysis** permet de gérer les pluriels (car/cars)
- **Capitalization** qui peut être juste le début de phrase ou un acronyme (led(forme passive de lead)/LED(diode))
- **Tokenization** qui permet de séparer les mots, ce qui peut devenir difficile ex: les mots composés.

Tout ceci n'est que les méthodologies intéressantes pour l'anglais, d'autres langages comme le japonais peuvent engendrer d'autre problème.

Le japonais a :

- Kanji
- Katakana
- Hiragana
- Romaji

De plus, trouver la fin d'une phrase est un challenge puisqu'un point ne veut pas forcément signifier la fin d'une phrase et peut être utilisé dans une abréviation.
Ce problème utilise donc un arbre de décision qui regarde différent features:

- La ponctuation
- Le format
- La police d'écriture
- Les espaces
- Les majuscules … etc.

Structure des données

Les données d'apprentissage peuvent être n'importe quel type de texte. Il suffit de pouvoir le lire depuis un programme informatique. Généralement après la lecture du texte celui-ci est traité, par exemple si on analyse une page HTML, il faut enlever les balises (<li>,<a>...). Le but est d'avoir un texte brut.

Échantillon d'apprentissage et de test

Pour nos exemples, les échantillons d'apprentissages sont composés de synopsis de film, les synopsis sont lus depuis un fichier texte. Le fichier contient une liste de synopsis séparés par des balises "NEXT" ou "BREAKS HERE", ces balises servent de délimiteur pour faciliter la lecture en python.

Synopsis Top 100 imdb

Synopsis Star Wars

Synopsis James Bond

Les échantillons de tests sont des synopsis de film non appris par le modèle. Ils sont chargés dans des variables directement dans le code, puis le cluster auquel ils appartiennent est prédit par le modèle.

Pre-processing

Avant de pouvoir utiliser des techniques de classifications ou de clustering sur le texte, il est possible d'appliquer quelques techniques pour obtenir de meilleurs résultats.

Stop word

En recherche d'information, un mot vide (stop word, en anglais) est un mot qui est tellement commun qu'il est inutile de l'indexer ou de l'utiliser dans une recherche. En français, des mots vides évidents pourraient être « le », « la », « de », « du », « ce », …

Ces mots sont donc enlevés pendant la phase de pré-traitement du texte.

Dans nos exemples, nous utiliserons la bibliothèque *sklearn* qui comprend une liste des stop words anglais.

Exemple

Phrase avec stop words :

Three years after the destruction of the Death Star, the Rebels are forced to evacuate their secret base on Hoth as they are hunted by the Empire…

Même phrase sans les stop words :

Destruction death star, rebels forced evacuate secret base Hoth hunted empire…

Stemming

En linguistique, la racinisation ou désuffixation (stemming, en anglais) est un procédé de transformation des mots en leur racine.

La racine d'un mot correspond à la partie du mot restante une fois que l'on a supprimé son (ses) préfixe(s) et suffixe(s).

Exemple

"fishing", "fished", "fish" et "fisher" deviennent "fish"

Notre phrase:

Destruction death star, rebels forced evacuate secret base Hoth hunted empire…

Devient :

Destruct death star rebel forc evacu secret base hoth hunt empir...

Grâce au stemming on peut comprendre que 2 mots différents partage la même racine et donc qu'ils font a priori parti du même lexique.

A l'inverse, le stemming peut aussi être source d'erreur. Par exemple en anglais, les mots "university" et "universe" ont la même racine ("univers"), alors que le rapport entre les deux peut être difficile à percevoir.

A noter que le stemming est un traitement final sur les mots, plus aucun traitement ne peut être effectué après.

TF-IDF: Term frequency-inverse document frequency

Le TF-IDF est une méthode statistique permettant d'évaluer l'importance d'un terme contenu dans un document, relativement à un corpus. Le poids augmente proportionnellement au nombre d'occurrences du mot dans le document. Il varie également en fonction de la fréquence du mot dans le corpus. Si le mot est souvent présent dans le document, mais pas dans le corpus son poids augmente en vice-versa.

Pour avoir une TF-IDF, on commence par compter le nombre d'occurrences de mot par documents. Par exemple :

	databases	huge	image	permanently	store
Document 1	1	2	1	0	0
Document 2	1	0	2	1	1
Document 3	1	0	2	0	1
Document 4	3	0	6	0	3

On peut grâce à cette matrice calculer l'IDF, qui mesure l'importance d'un terme dans un corpus :

$$idf = log10(\frac{nombre\ de\ documents\ dans\ le\ corpus}{nombre\ de\ documents\ où\ le\ terme\ apparaît})$$

	databases	huge	image	permanently	store
IDF	0	0.602	0	0.602	0.125

Comme databases et image apparaissent dans tous les documents, ils ne sont pas utiles dans la différenciation des documents, comme le montre le calcul de l'IDF.

Grâce à ces 2 matrices, nous pouvons calculer la TF-IDF qui correspond à la multiplication des 2 :

	databases	huge	image	permanently	store
Document 1	0	0.602	0	0	0
Document 2	0	0	0	0.602	0.125
Document 3	0	0	0	0	0.125
Document 4	0	0	0	0	0.375

La TF-IDF d'un terme est élevé quand le terme apparaît beaucoup dans le document, mais est rare ailleurs.

On peut également prendre en compte certains paramètres, si un mot est présent dans plus de 80% des documents, on supposera qu'il n'est pas important donc il n'entrera pas en compte dans les calculs. De même, si un mot est présent dans moins de 20% des documents, le mot sera trop spécifique donc n'apportera pas d'information pertinente.

K-MEANS

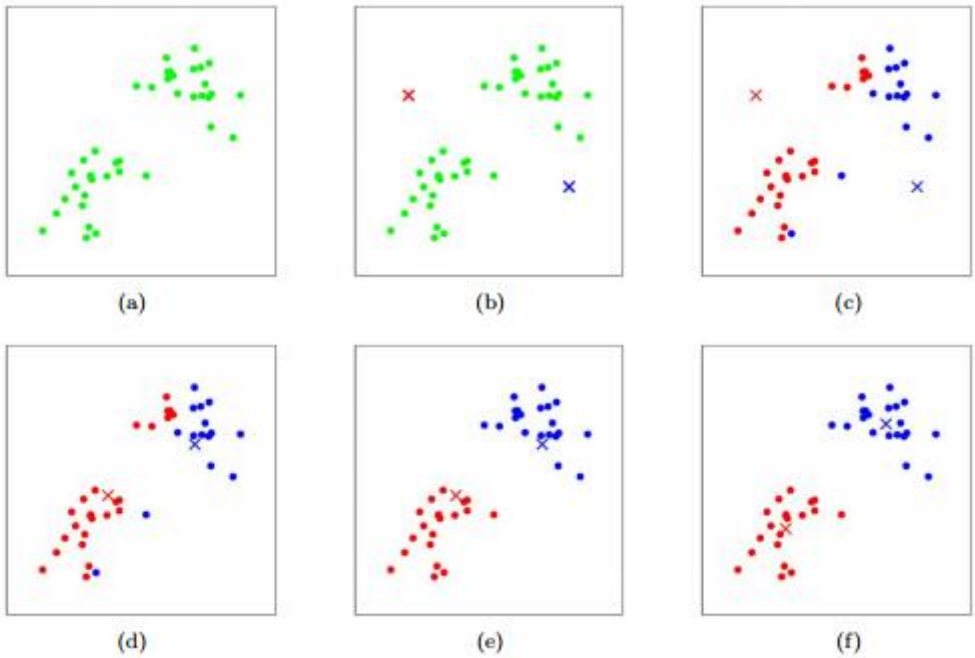
Une fois les étapes de Pré-Traitement et le calcul de la matrice TF-IDF effectué, il est possible d'appliquer des méthodes de classification et de clustering, k-means sera utilisé ici.

K-means doit être initialisé avec un nombre prédéterminé de cluster (k), il existe des méthodes automatiques pour déterminer ce nombre, (elbow method par exemple) qui ne seront pas décrites ni utilisées ici.

L'objectif est de segmenter les données en k groupes, à l'initialisation de l'algorithme, on place k points (centres) dans l'espace, à chaque centre sont associés les données qui lui sont proches : cela crée un groupe autour de chaque centre.
Ensuite, on calcule le centre de gravité de chacun de ses groupes : ces k centres de gravité deviennent les nouveaux centres et on recommence tant que les groupes ne sont pas stabilisés

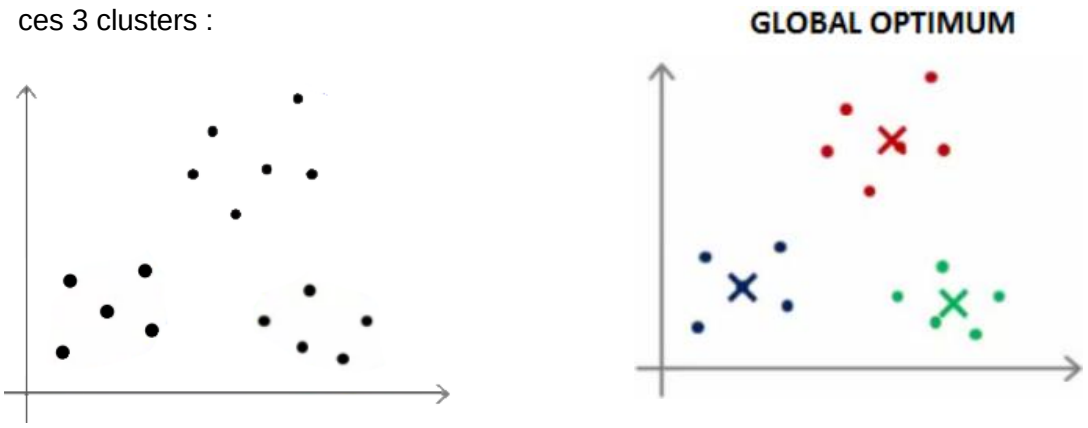
La stabilité signifie qu'il n'y a pas de données qui changent de groupe d'une itération à la suivante. Ou encore l'inertie ne varie pas substantiellement d'une itération à la suivante.

Exemple d'itération de k-means :

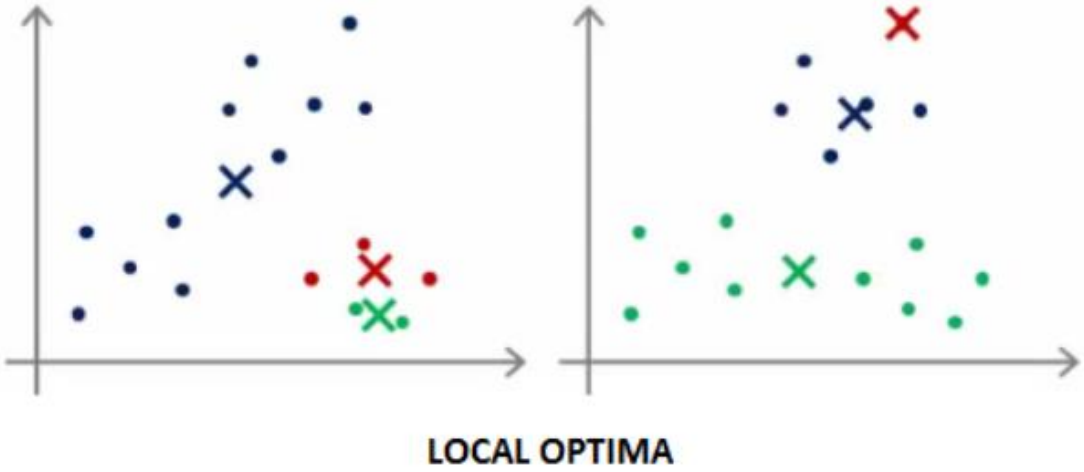


Les centres de gravité des clusters étant placés aléatoirement au départ chaque itération de k-means donnent un résultat différent, en effet dans notre cas k-means tombe souvent sur un maximum local.

Par exemple, les points ci-dessous devraient donner ces 3 clusters :



Mais on peut tomber dans un des deux cas suivants :



Il faut donc exécuter plusieurs fois l'algorithme pour converger vers un optimum global.

Résultats

Le premier exemple consiste à trier en 2 clusters des films “James Bond” et “Star Wars”. Nous envoyons les synopsis de ces films au programme pour les analyser, sans préciser quel synopsis correspond à quel film.

A la fin nous affichons les mots les plus courants par clusters :

Cluster 0 : bond british kill investig face

Cluster 1 : luke vader obi wan jedi

A première vu les clusters semblent corrects, nous allons donc les utiliser pour prédire le cluster de deux nouveaux synopsis :

```
Y = vectorizer.transform(["""About 30 years after the destruction of the Death
                          Star II, Luke Skywalker has vanished following the
                          demise of the new Jedi Order he was attempting to
                          build."""])

prediction = model.predict(Y)
print(prediction)

Y = vectorizer.transform(["""After an operation in Istanbul ends in disaster,
                          Bond is missing and presumed to be dead. In the
                          aftermath, questions are raised over M's ability to
                          run the Secret Service, and she becomes the subject
                          of a government review over her handling of the
                          situation."""])

prediction = model.predict(Y)
print(prediction)
```

```
Top terms per cluster:
Cluster 0:  luke  vader  wan  obi  jedi
Cluster 1:  bond  british  kill  investig  face


Prediction
[0]
[1]
```

Le premier synopsis est celui d’un film Star Wars est prédit dans le cluster 0 qui est celui en rapport avec Star Wars. La prédiction est correcte, elle l’est aussi pour le second synopsis, celui d’un film James Bond.
Dans le second exemple, nous utilisons les synopsis des 100 meilleurs films selon Imdb (Internet Movie Database).

Après plusieurs tests, nous avons décidé de les classer en 6 clusters, voici les résultats :

Cluster 0: order kill men attack offic

Cluster 1: say ask polic look goe

Cluster 2: father death polic murder meet

Cluster 3: famili kill discov head father

Cluster 4: love return life war marri

Cluster 5: fight hand person chang see

Nous pouvons observer que certains mots sont coupés dû au stemming effectué précédemment.

Les clusters pourraient s'assimiler à des genres de films :

Cluster 0 : Action

Cluster 1 : Policier + Famille

Cluster 2 : Policier

Cluster 3 : Drame

Cluster 4 : Amour + Guerre

Cluster 5 : Combat

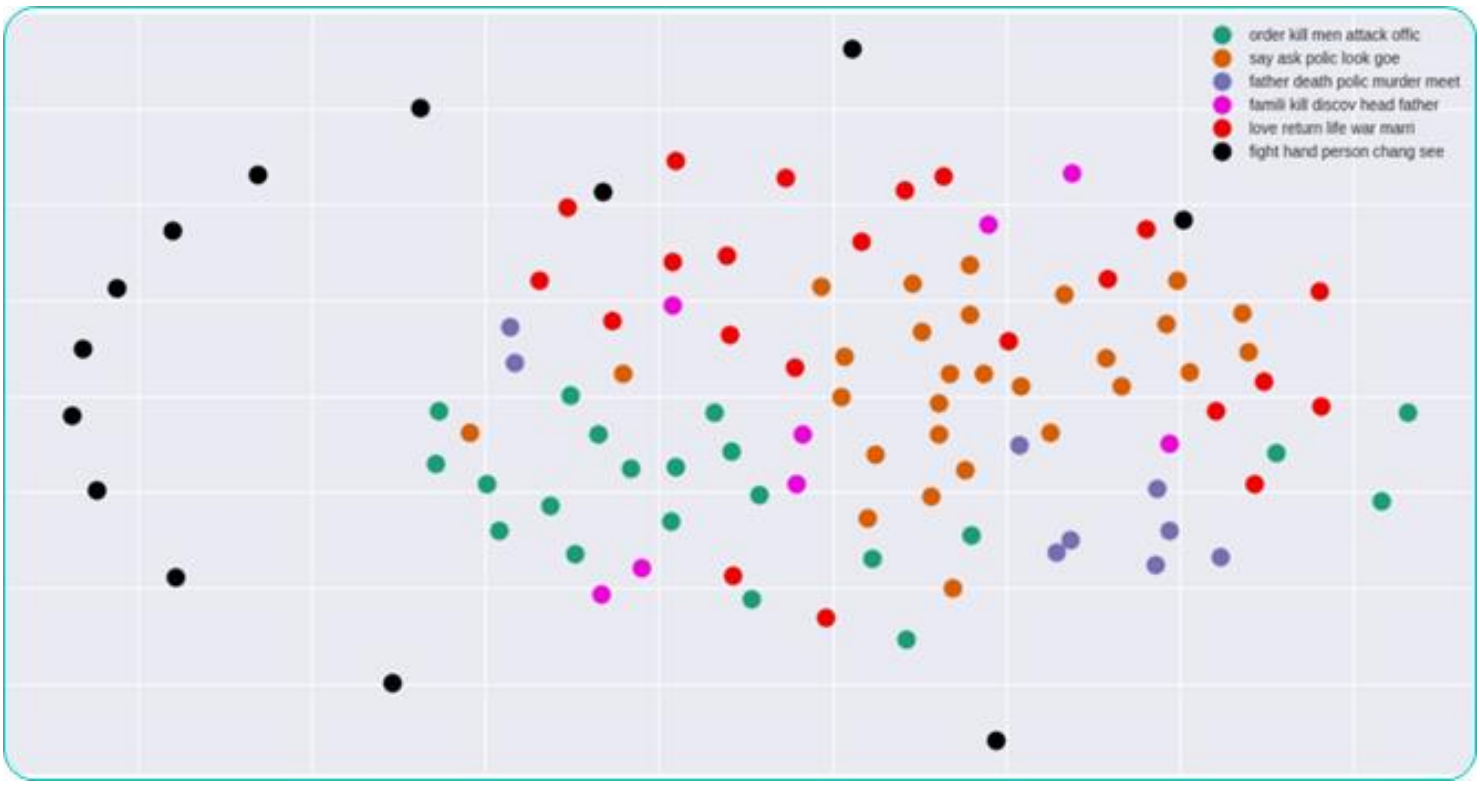
Visualisation des résultats

K-means travail avec la matrice TF-IDF qui a autant de dimensions qu'il y a de mot différent dans les textes. Le résultat retourné comprend donc autant de dimensions. Pour pouvoir afficher quelque chose, il faut convertir ce modèle en X dimensions en un modèle à 2 ou 3 dimensions.

Nous utilisons la méthode de positionnement multidimensionnel (Multidimensional scaling, en anglais), cette méthode permet de garder la même distance entre deux points d'une dimension à une autre. Si deux points sont séparés par une distance *a* dans un espace à six dimensions, ils resteront à une distance *a* dans un espace à deux dimensions.

Voici ce que nous obtenons pour le premier exemple :

Et pour le second :



Technologies utilisées

- Google Colaboratory https://colab.research.google.com/

Colaboratory est un projet de google donnant accès à un environnement Jypyter notebook (exécution de code python). Cela ne requiert pas de configuration ni d'installation. Le code est exécuté dans le cloud.

Les notebooks sont enregistrés dans google drive, ils peuvent donc être partagés facilement.

- sklearn http://scikit-learn.org/stable/

Sklearn est une librairie python destinée au machine learning, elle contient des algorithmes de classification, regression, clustering...

- nltk http://www.nltk.org/

Ntlk propose différents outils pour développer des programmes python analysant le langage humain (Stop words, bibliothèque de stemming…).

Conclusion

Le text-mining est un pan du forage de données qui est intéressant et prometteur pour l'avancer d'extraction de connaissances. Sa propagation améliorera la productivité des entreprises. Elle pourra aussi amener à une analyse des tendances sociétale par les flux de données que sont des réseaux sociaux.

Nous avons aussi pu montrer la puissance du text-mining malgré les techniques basiques utilisé dans notre exemple. Nous avons pu associer différents films entre eux avec un minimum d'analyse de texte sans prendre en compte la relation ou le sens des mots.