# Disclaimer

Bonjour, vous êtes dans ce Word un peu comme dans ma tête. Il se peut que vous repartiez avec plus de questions que de réponse. Mais ça peut aussi très bien se passer. J’ai essayé d’organiser le document afin qu’il soit plus lisible aux hommes du dehors (comme vous).

Table des matières

[Disclaimer 1](#_Toc105753013)

[Analyse de l’article donnée 1](#_Toc105753014)

[Introduction de l’article 1](#_Toc105753015)

[Calcul de la longueur d’un topic 5](#_Toc105753016)

[Evolution journalière 5](#_Toc105753017)

[01/06/2022 5](#_Toc105753018)

[03/06/2022 6](#_Toc105753019)

[07/06/2022 9](#_Toc105753020)

[08/06/2022 11](#_Toc105753021)

[09/06/2022 14](#_Toc105753022)

[10/06/2022 22](#_Toc105753023)

# Analyse de l’article donnée

Page article : <https://arxiv.org/abs/2010.01600>

On utilise un modele de « topic » (topic modeling) qui compare des topics qu’on identifie comme soit :

* Long-lasting trends
* Short-lasting topic

Le modèle doit aussi savoir localiser ses topics dans le temps avec clarté.

Cet article présente différents modèles.

On compare la « variabilité » (variability) des tailles/longueur de divers topics.

Ces longueurs étant celles obtenues en utilisant différents modèles (de topic) assez connus :

* Latent Dirichlet allocation (LDA)
* Nonnegative matrix factorization (NMF) (sujet de mon stage)
* Nonnegative CANDECOMP/PARAFAC tensor decomposition (NCPD et Online NCPD)

Le NCPD est censé être la « tensor counterpart » de la NFM. Cela veut surement dire que c’est des modèles similaires mais que le NCPD utilise les tensor contrairement au NFM. ( ?)

L’article veut nous montrer que seules les méthodes utilisant des tensor arrivent à détecter les short-lasting topics (topics de courte de durée) avec succès.

De plus, les méthodes utilisant les tensors sont plus précises (accurate) pour connaitre le moment d’apparition et de disparition d’un topic. Elles sont plus précises que la LDA et le NMF.

L’article propose de faire des mesures « quantitatives » de la taille des topics. Cela servira à illustrer la performance du NCPD lorsqu’il faut trouver/découvrir des topics. Les topics utilisés seront pour certains « semi-synthetic » ou issues de données réels (real-world data). Les données comprennent des intitulés de « news » (news headlines) ainsi que des tweets liés au covid-19.

Les données « semi-synthetic » sont surement des données en partie inventées. Peut être à partir d’autre données réelles ?

## Introduction de l’article

Quel est le format des données ?

Modèle LDA :

* Modélise les topics ou les classe par probability de distribution de « set de mots »(set of words).
* Ces sets de mots évoluent selon un « scheme Bayesien » (Bayesian scheme) en donnant au « batches » une matrice « words x documents ». C’est surement un tableau contenant le nombre de fois qu’un mot apparait pour chaque document.
* Il semble qu’on reçoit en sortie deux « représentations » de type « words x topics » et « topics x documents). Cela veut surement dire qu’on a en sortie deux autres matrices créées par notre modèle. Le modèle à du créer lui-même les « topics » en analysant les mots par documents. Puis il a dû faire des associations et compter le nombre de topics par document ainsi que la relation entre un mot et un topic identifié.

Modèle NMF :

* C’est aussi un modèle qui utilise des matrices (comme le LDA). Il va prendre la matrice « words x documents » et la « décomposera » en « words x topics » et « topics x documents »
* Les documents peuvent avoir un « timestamp ». C’est surement un fichier ou un marquage qui permet d’indiquer le moment ou est publié le texte. Si le « timestamp » est ordonné en fonction du temps, la matrice « topics x document » pourra trier et présenter les topics dans cet ordre.

Il y a deux façons de faire que le NMF prenne en compte les « tranches » (slices) temporelles simultanées ou simultanément

* On peut factoriser chaque tranche temporelle

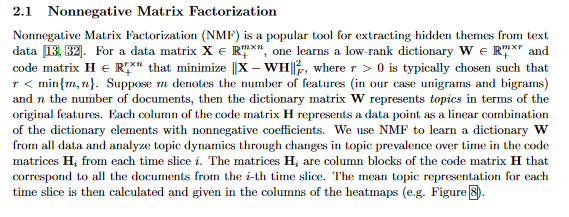
“There are two basic methods of using NMF for dynamic topic modeling in a way that accounts  
to the time slices that are essentially simultaneous for the considered dynamic. First, one can  
factorize each time slice independently using NMF and obtain topic structure for each time slice  
[35, 38, 3, 36]. Second, one can concatenate all the time slices along the documents dimension and  
decompose the resulting matrix using NMF with a fixed dictionary to obtain common topics. The  
evolution of such topics is then found by computing their contribution in each time slice [18]. In  
this work, we do the latter to post-process both NMF and LDA for better visualisation and more  
interpretable results on the time scale. We discuss the methods in more details in Section 2. »

Concatenate all the time slices”

On peut encoder le corpus de documents dans un tensor à 3 dimensions

Qu’est-ce que le entrywise Froebenus norm ?

unigrams and bigrams



Comment fait l’algorithme pour extraire des « hidden themes » d’un texte ?

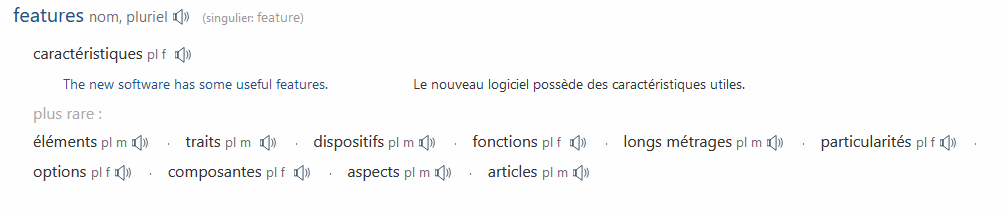
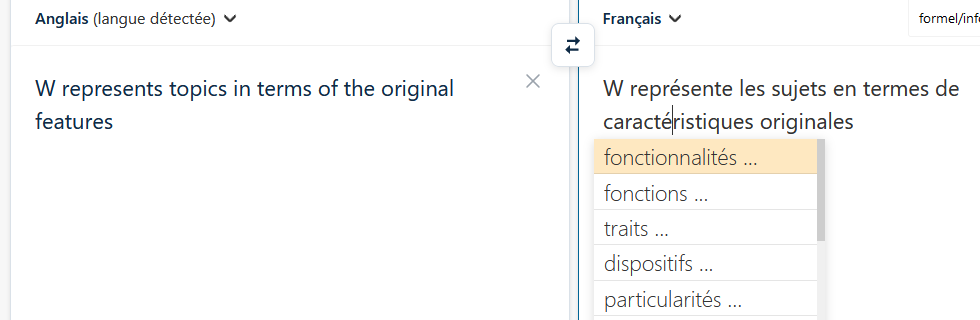
Il devrait pouvoir :

* Savoir ce qu’est un themes
* Savoir le trouver dans un texte
* Savoir l’extraire

Qu’est-ce qu’un « low rank dictionary W » ainsi qu’un « code matrix H » ?

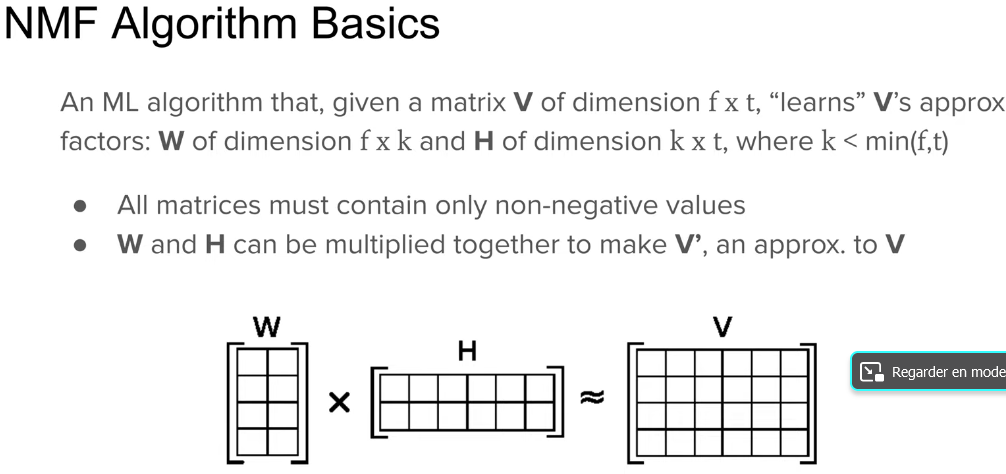
* Comment obtient-on ces deux matrices à partir de la première ?
* Quelle sorte d’information contiennent ces deux nouvelles matrices ?

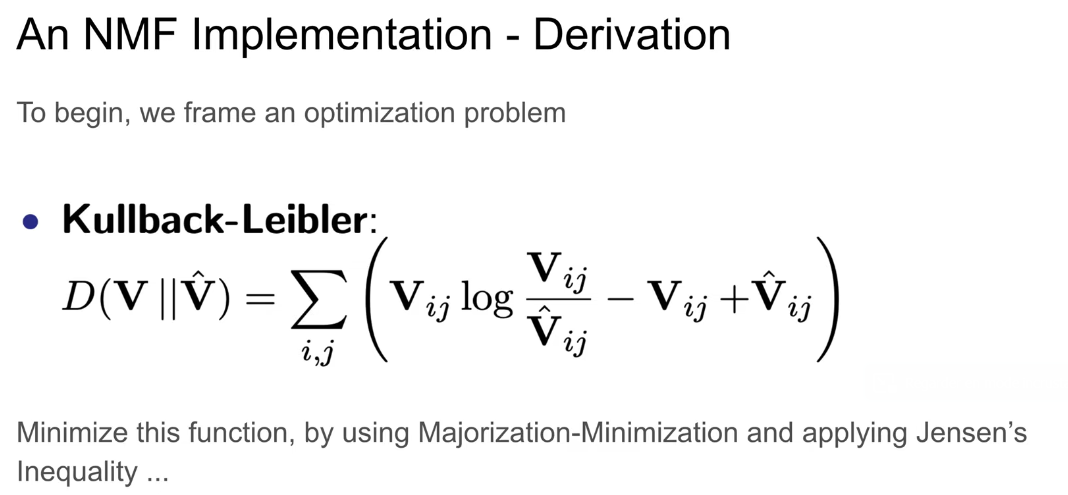
W contient les topics qui ont été repérés par notre modèle dans une étape précédente. W est de dimension m\*r. « m » étant le nombre de « features » .« r » étant choisi de sorte à être plus petit que m (nombre de features) et n.

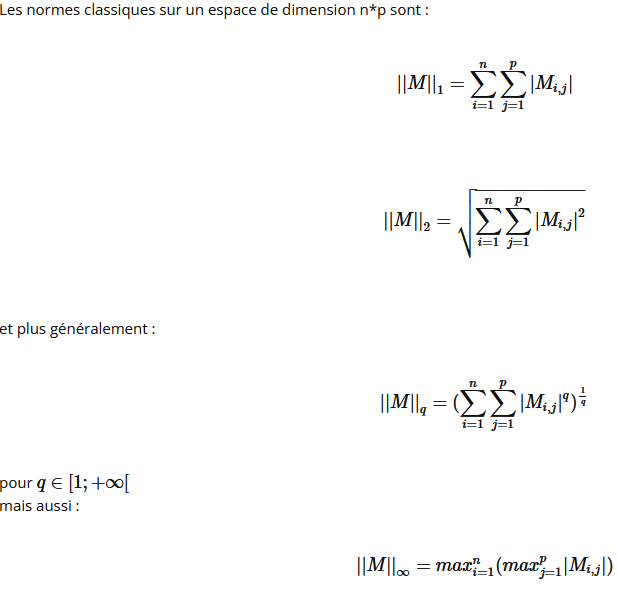
* Qu’est-ce qu’un « feature » ? On nous dit qu’ils sont ici sous forme de « unigrams and bigrams ». On nous dit aussi que W représente les topics sous forme de « original features ».
* 
* 
* J’en déduis que notre modèle après avoir trouvé les topics leur a donné des noms « features » composés d’un ou deux mots.
* 
* Wi,j représente le nombre de fois qu’un mot est utilisé dans un topic i apparait en fonction de son rang j. Soit Wi,j contient un feature. Le nombre de mot r étant
* Il semble que l’ensemble de ces matrices doit vérifier une condition, ils doivent minimiser la distance au carré entre X et WH. Je ne suis pas
* W associe à chaque topic un feature (unigram ou bigram), chaque Wi,j contient le nombre de fois que cette association est faite.
* Chaque colonne de H représente une donnée comme une combinaison linéaire d’élements du « dictionnary ».

Qu’est-ce que sont les unigrams et bigrams ?

“one learns a low-rank dictionary W” – on fait apprendre cette matrice à notre modèle ?



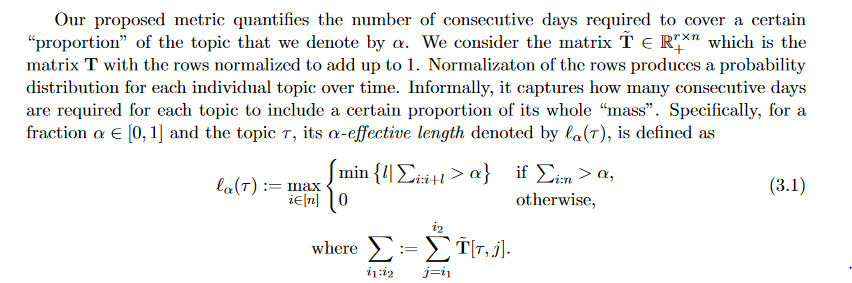




<https://www.youtube.com/watch?v=_QiTQQDrx5I>

<https://www.youtube.com/watch?v=ZspR5PZemcs>

## Calcul de la longueur d’un topic



# Evolution journalière

## 01/06/2022

Recherchons dans le document des informations sur comment se fait la décomposition de la matrice initiale en deux matrices.

Rappelons qu’il faut faire plusieurs fois cette opération.

La recherche dans l’article n’a pas été fructueuse. On nous dit qu’on créé les matrices W et H mais je ne sais pas exactement comment.

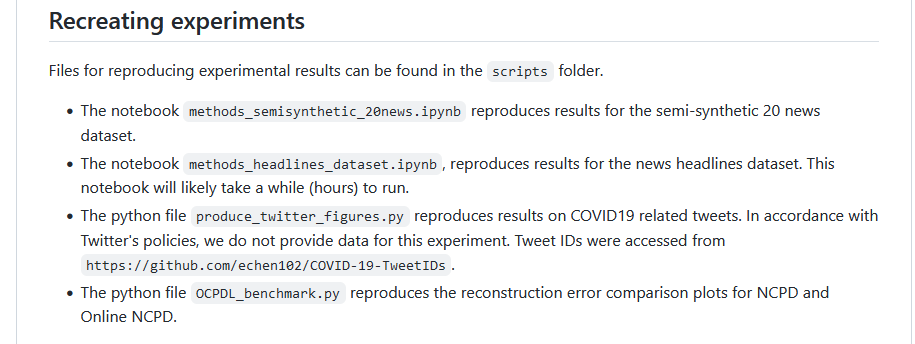
Regardons plutôt les ressources git à disposition :

* Le repo où déposer nos fichiers
* Un repo git où se trouvent le travail de Hanbaek lyu et Lara kassab sur le topic modelling
* Ainsi qu’un « notebook » qui contient un code de programme pour faire du NMF, LDA, NCPD et ONCPD.

J’ai décidé d’exécuter le code de Lyu et Lara dans un premier temps.

J’ai tout d’abord effectué une copie du repo qu’ils proposent en local puis il a fallu ajouter les données « ABC news ».

Ouvrir jupyter notebook et suivre les consignes du README.

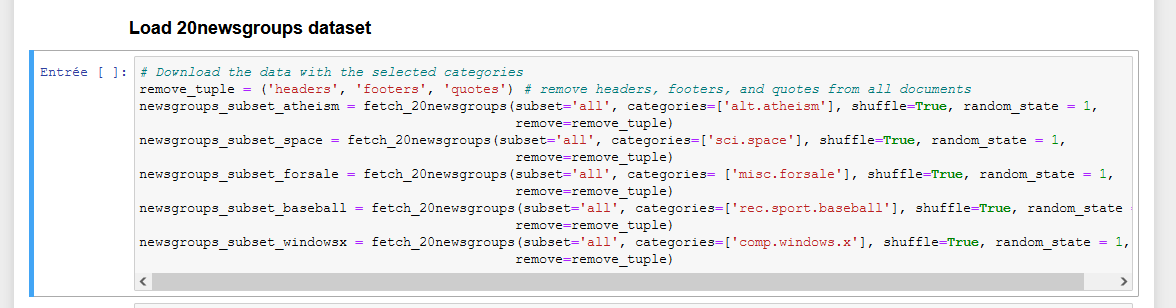


Cette partie du readme mis à disposition va nous faciliter la tache afin de tout exécuter. Nous pouvons ouvrir chacun de ces notebooks afin d’essayer une ou plusieurs méthodes de topic modeling sur des données différentes.

Voyons comment ça se passe

On décide d’exécuter le notebook « methods\_semisynthetic\_20news.ipynb »

Voyons ce qu’il fait par étape



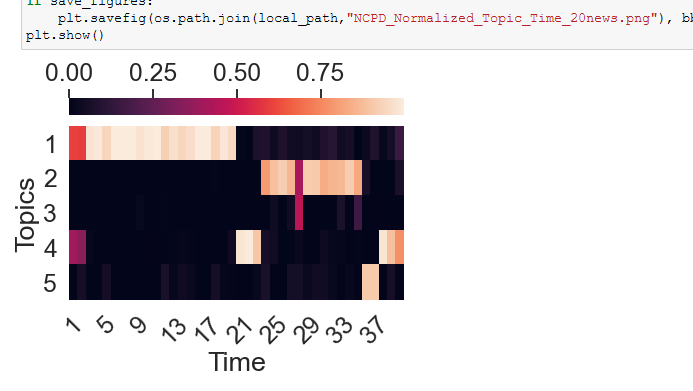
Il charge les données mais ne les fait pas d’un coup, on dirait qu’il y a une sorte de découpage et de réorganisation.

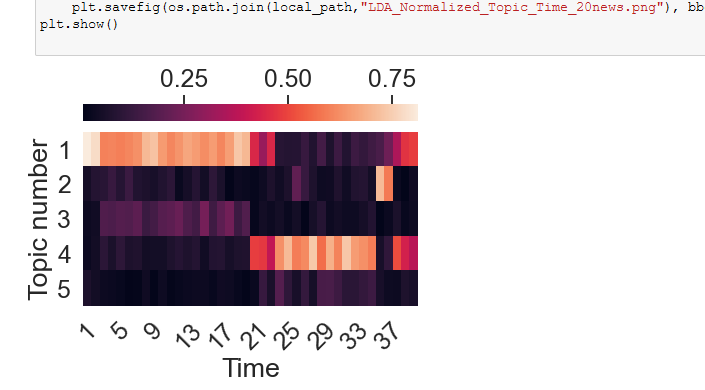
Ah on dirait qu’il ne charge pas tout le dataset mais uniquement certaines colonnes. Mais je ne sais pas en détail ce qu’il fait.

## 03/06/2022

Nous avons pu continuer avec le même code et lancer une NMF, NCPD et LDA

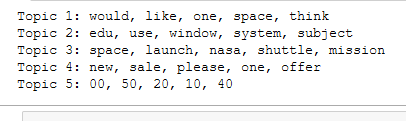






Mystères :

* La façon de découper et réorganiser les données en matrices et tensors
* La façon de choisir les topics (exemple LDA)



* La structure de départ des données

En résumé le code a eu un peu de mal à s’exécuter. Il y avait des bibliothèques à installer ainsi que quelques variables dont les noms devaient être un peu modifiées.

Essayons d’exécuter l’ONCPD en analysant de plus près les parties ou alors en montrant ce que je ne comprends pas

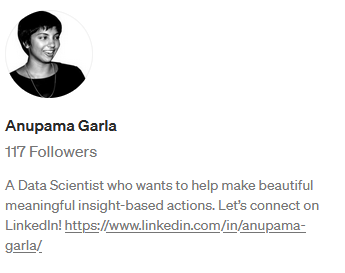
Les étapes de l’ONCPD dans le programme :

* # SVD intialization (Optional)
* # Run Online NCPD
* # Obtain the shape of the factor matrices
* # Display topics
* #Visualize topic distributions

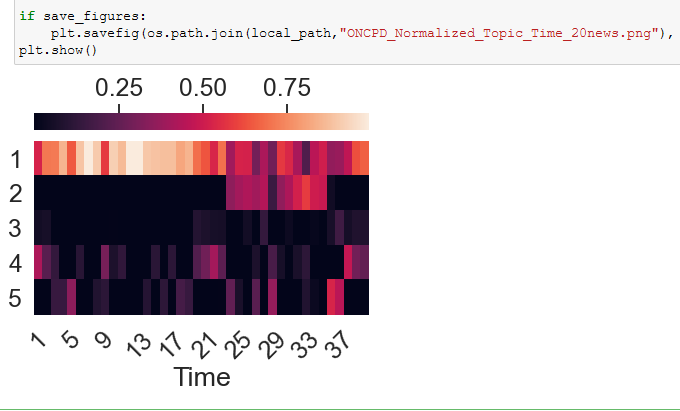
J’ai tenté de trouver une ressource expliquant la nmf et je suis tombé sur ce site :

<https://towardsdatascience.com/nmf-a-visual-explainer-and-python-implementation-7ecdd73491f8>

C’est beau. C’est de : Anupama Garla

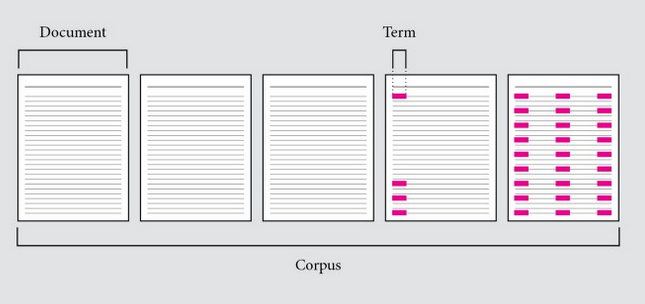


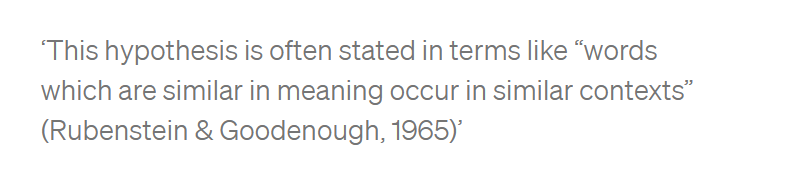
Sinon voici ce que me rend l’ ONCPD

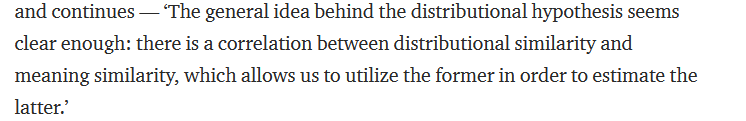


## 07/06/2022

Je suis de retour et nous allons lire l’article de anupama. Aujourd’hui j’ai malheureusement oublié ma trousse chez moi, ça ne va pas du tout. Passons.

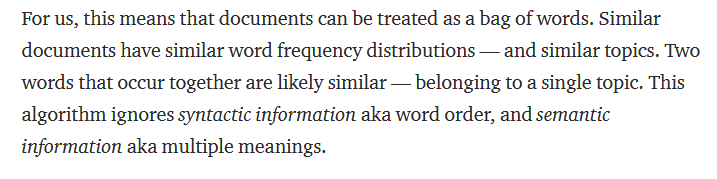


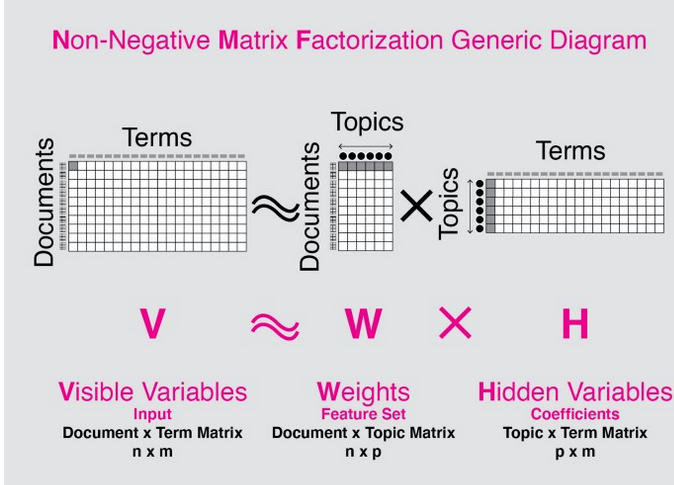




Mais c’est trop bien cette théorie.

On peut donc détecter un topic à l’aide de la distribution des termes le long d’un document ?

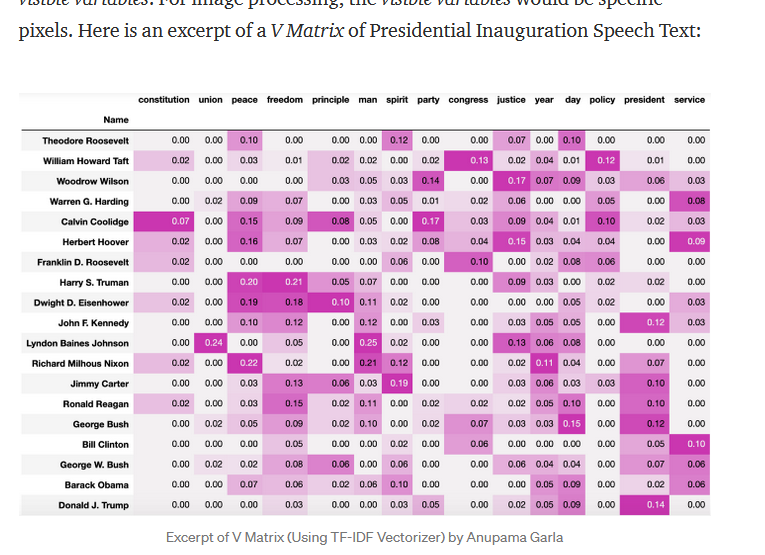




Le retour de la mystérieuse décomposition. Ce qui est mystérieux est la méthode exacte de « décomposition ». Puis comment trouve-t-on les topics ?

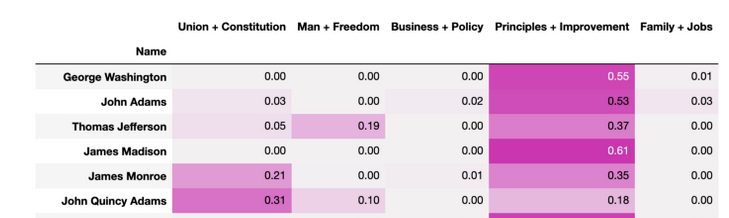
J’imaginais jusque là qu’une sorte d’analyse permet de les trouver.

Ce que je sais, c’est qu’il faut en trouver r.

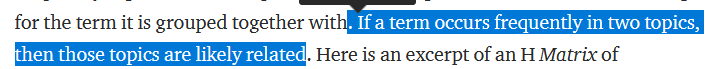


Voila une chose surprenante. On a pour la première fois un exemple visuel de matrice qu’on prend en entrée pour la NMF. Cependant on y retrouve des nombres à virgule. Des proportions ?

On veut ensuite nous montrer la matrice W.



Là je reconnais les topics mis sous forme de bi-grams.

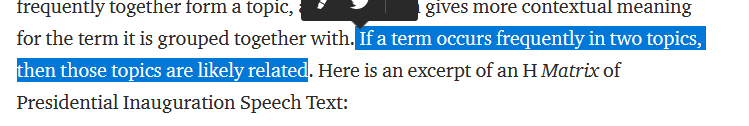


## 08/06/2022

Bonjour, aujourd’hui nous allons continuer à analyser ce bel article afin d’essayer de comprendre le maximum de choses.

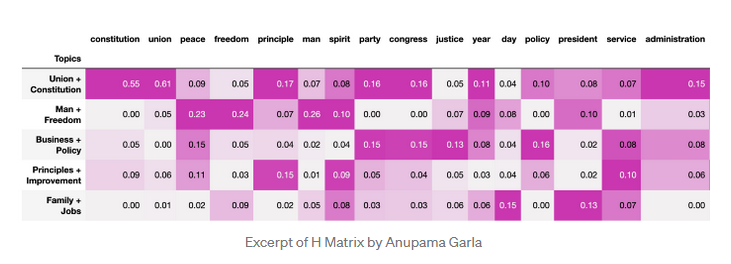
Je me demande comment ils calculent les probabilités qui composent nos matrices. Peut être que sur le nombre total de mots dans un document donné ils comptent combien de fois celui-ci apparait. Ce nombre est ensuite divisé par le nombre total de mot dans le document.





Imaginons que les termes banane, gâteau et ananas reviennent souvent ensemble. Ça voudrait dire que banane-gâteau

Forment un topic mais gâteau-ananas et banane-ananas aussi ? A partir de quel moment peut-on dire qu’un groupe de terme revient « souvent » ?

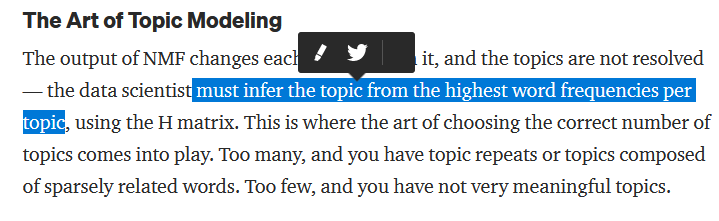


Autre questionnement : Dans ce tableau, comment obtient-on nos nouvelles valeurs de topic par mot ? J’ai une idée. Pour chaque topic, on regarde à travers tous les documents où il se trouve et on compte le nombre de fois où chaque mot apparait.

Parenthèse méthodologique : est-ce que j’avance assez avec ma façon de lire l’article, de faire des déductions puis de continuer la lecture ?

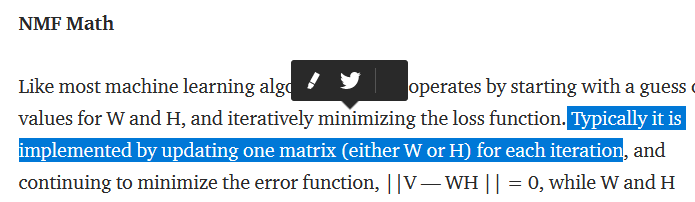
Bon je suis allé voir Luan qui m’a donné quelques explications sur l’exécution des codes python qu’on nous a donné. Il m’a parlé de l’exécution et de la nécessité des créer des environnements. C’est une étape que je ne comprenais pas en entier mais que je ne pensais pas nécessaire. J’ai pu exécuter entièrement le code des « 20news ».

Retournons voir notre article.



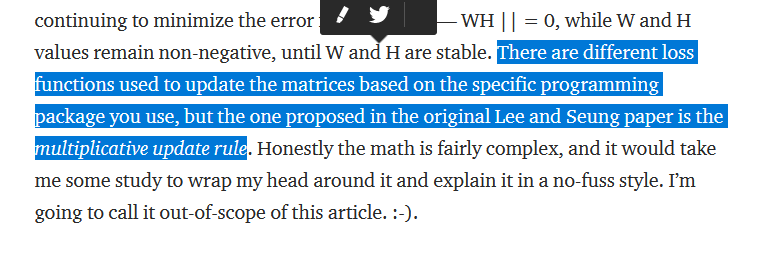
NMF. Le F veut dire factorisation. On fait donc une factorisation « non-négative » de la matrice de base. C’est ce qui permet d’obtenir les deux autres matrices ?

Il faut en tout cas choisir le nombre de topic pour en avoir assez mais pas trop en fonction de nos données.



Intéressant

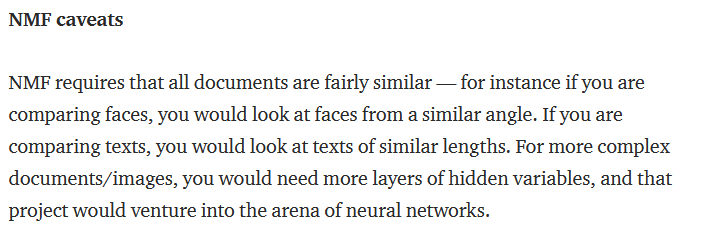
On met à jour l’une des matrices seulement pendant qu’on s’assure que cela diminue la « fonction d’erreur » (error function).



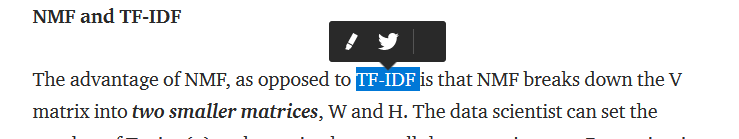
Très intéressant.

« loss functions » et « multiplicative update rule ».

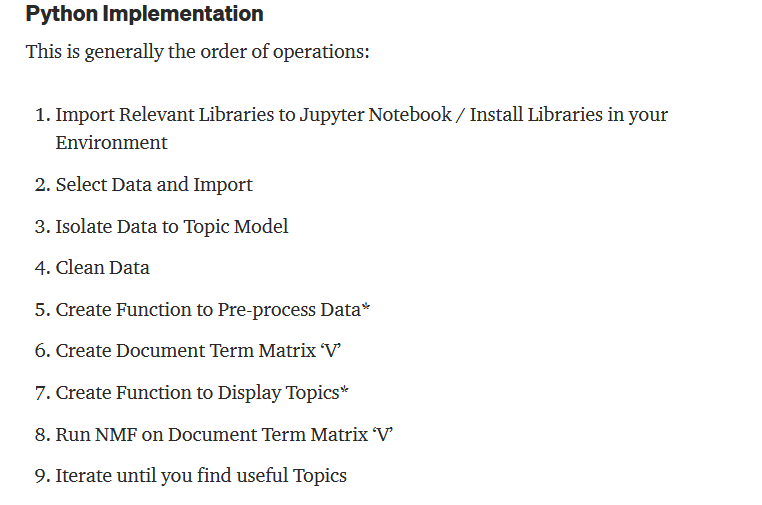
Je me demande si « loss function » c’est la même chose que « error function ».



Intéressant



TF-IDF est une sorte d’autre méthode de topic modeling ?



J’espère pour demain exécuter plusieurs codes et explorer cette histoire de directory de plus près.

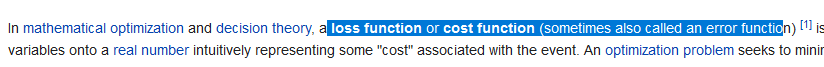
## 09/06/2022

Bonjour tout le monde.

Où en étions-nous ? Ah oui, les programmes à exécuter.

Bon je n’arrive pas vraiment à me concentrer là-dessus. Faisons quelque chose d’utile en attendant.

Bon bah commençons par voir ce qu’est une « loss function » finalement.



Donc la « loss function » et « error function » sont bel et bien les mêmes.

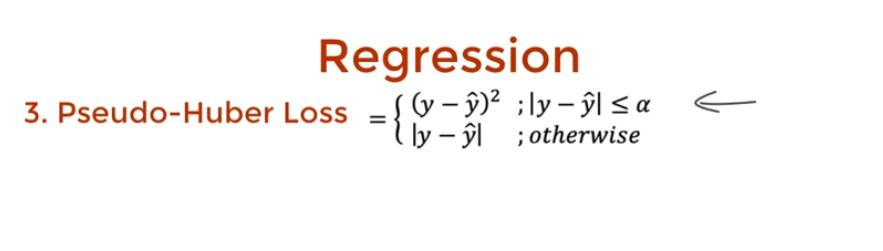


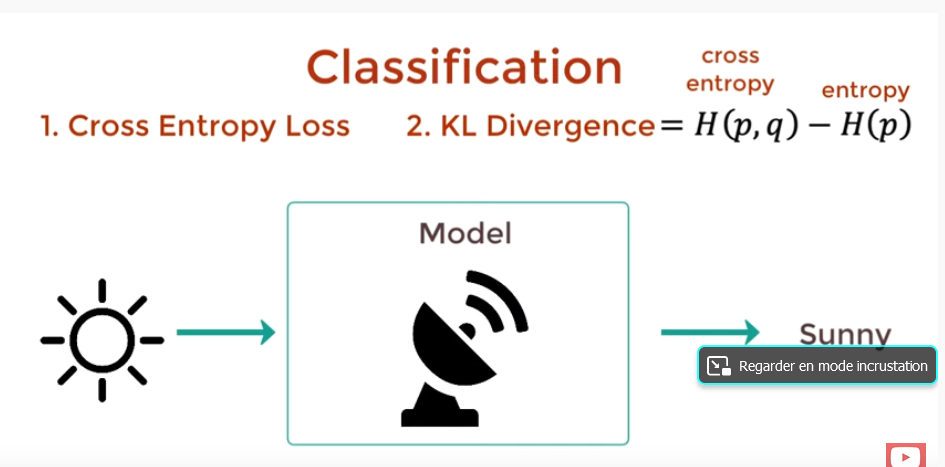
Aah il se peut que je l’aie déjà croisé quelque part cette fonction de perte puisque j’ai dû faire de l’optimisation.

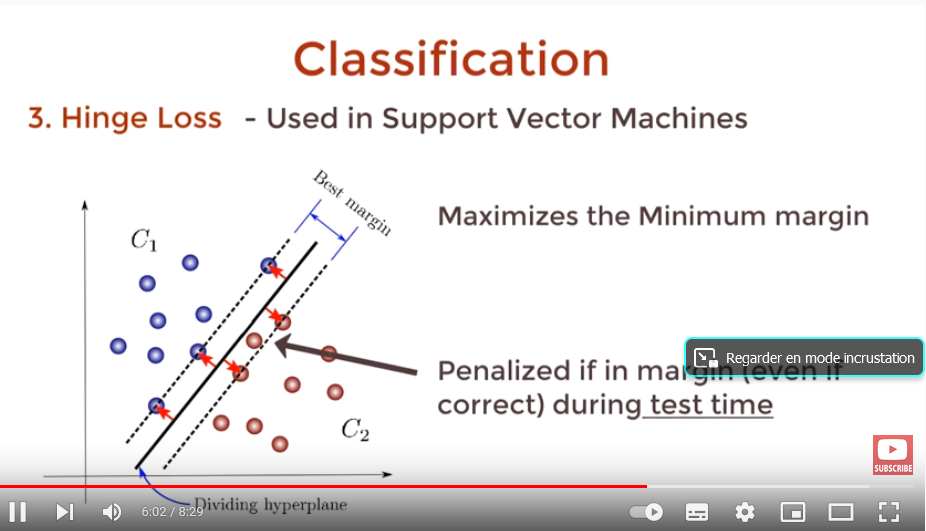


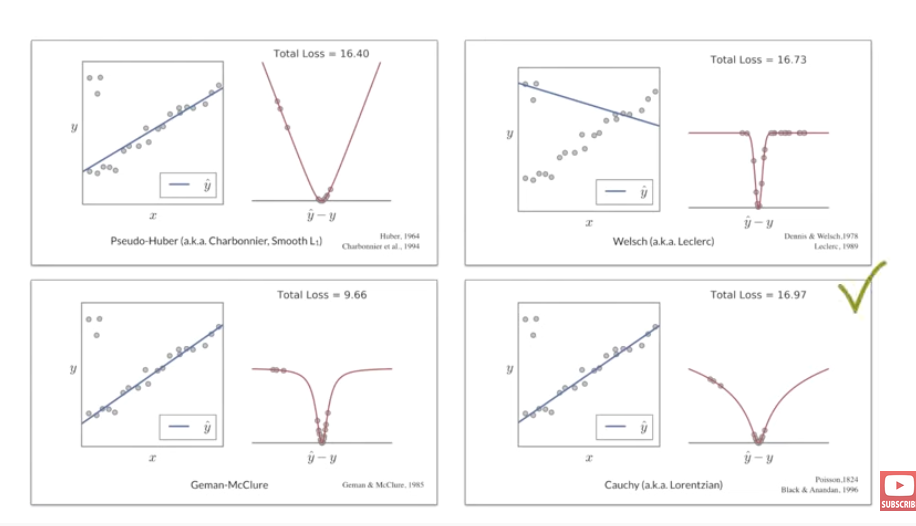
Mhhm, intéressant. C’est donc très utilisé.

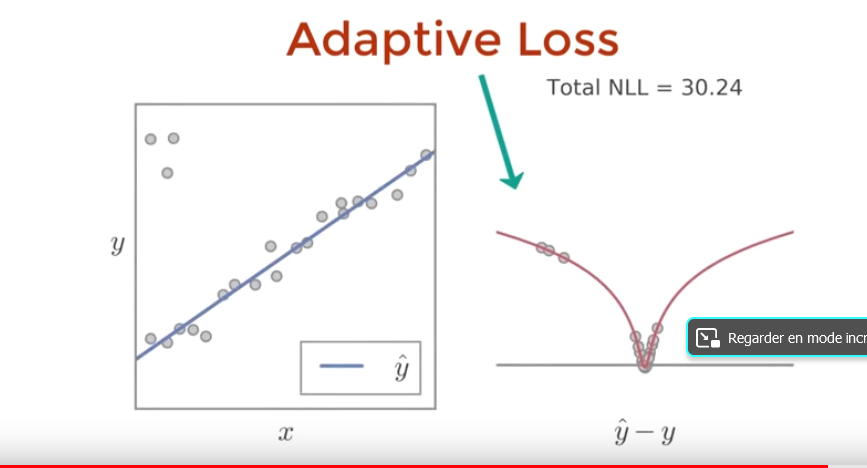
Je vais voir cette video tiens : <https://www.youtube.com/watch?v=QBbC3Cjsnjg>





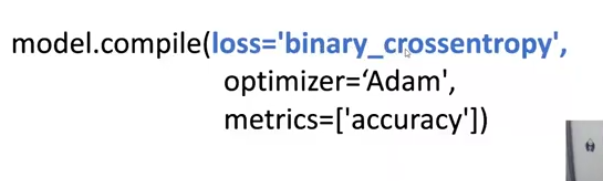




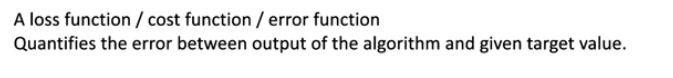


C’était très intéressant, on dirait des droits de regression.

Voyons maintenant cette video : <https://www.youtube.com/watch?v=-qT8fJTP3Ks>

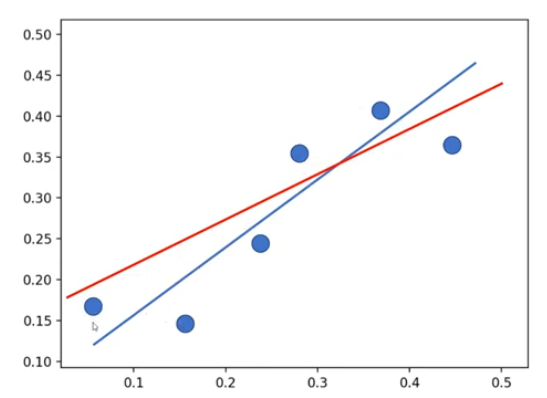


Beurk



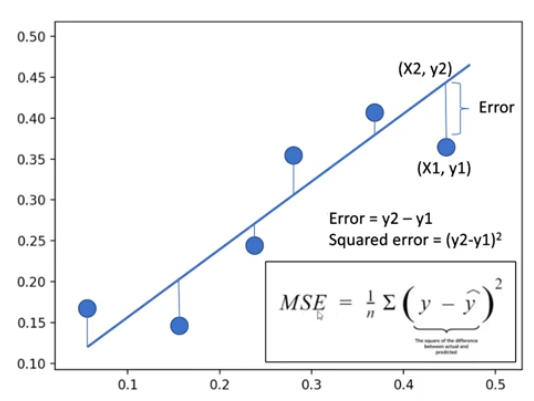
Il nous parle de training our model

Quantifies de error between the output and the expected value



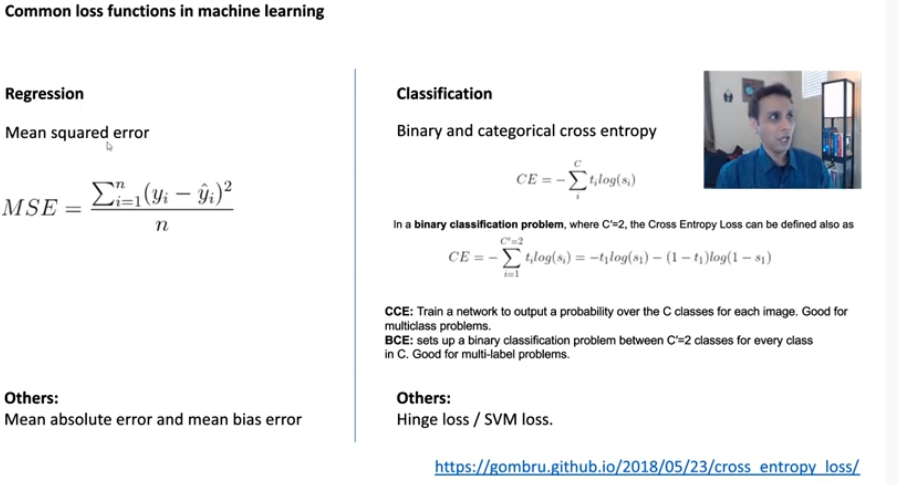
Le rouge prédit mieux certaines données

Le bleu en prédit mieuc d’autres



Square it positive/negative side don’t matter

Mse mean squared error



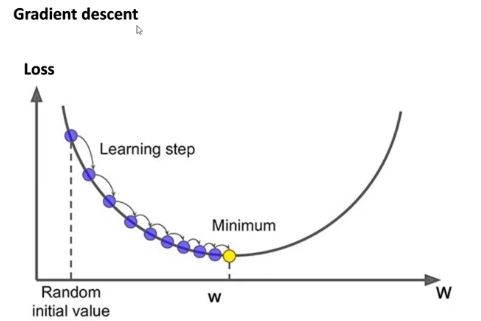
Une autre video : <https://youtu.be/aJToUocPLg4>

Très intéressant mais je n’ai pas retenu grand-chose.

Et une autre encore : <https://youtu.be/JhQqquVeCE0>



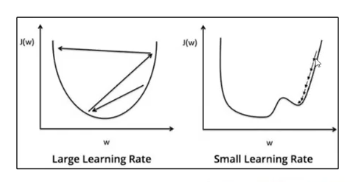
Comme en modèle linéaire/régression linéaire



Next step à gauche ou droite puis calcul de loss function pour savoir si on va dans la bonne direction pour trouver le minimum

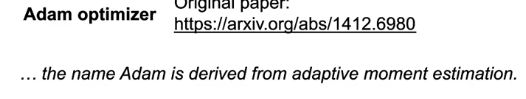
Faire des grand ou petit pas

Learning rate = la taille des pas



Alors ça c’est drole, tu peux te tromper de minimum

Pseudo-minimum



Mhhhm interessant

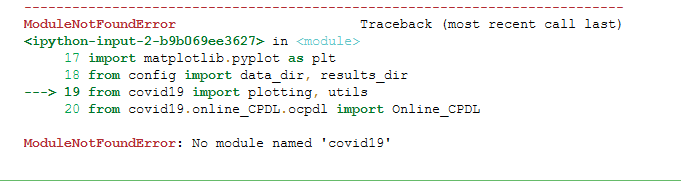
Passons maintenant à l’execution de « methods\_headlines\_dataset » :



Cette fois ci j’ai essayé de faire :

conda env create -f environment.yml

Car j’avais cette erreur :



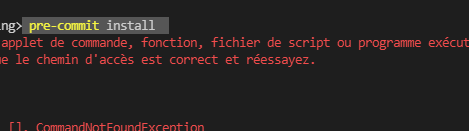
Il semble que ce ne soit pas fonctionnel après cette simple commande même s’il semble m’avoir installé plein de choses étranges. Des librairies je crois :



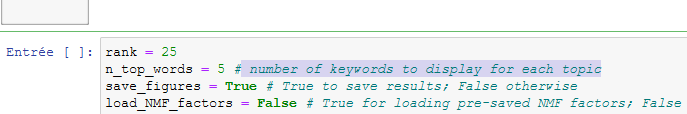
Voilà à quoi ça ressemblait.



Ajoutons ceci pour voir



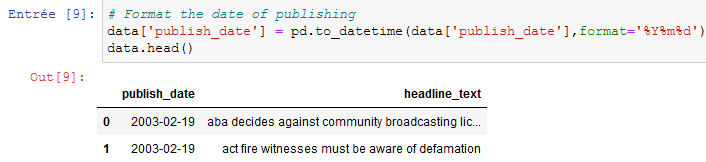
La chose ci-dessus ne fonctionne pas vraiment



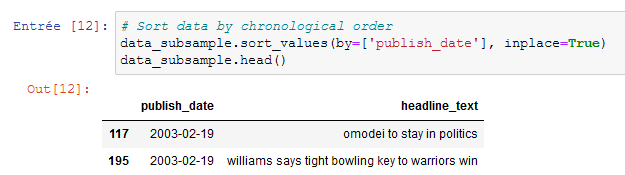
Intéressant. On retrouve quelque chose qui ressemble à ce qu’on a vu sur le site de Garla.



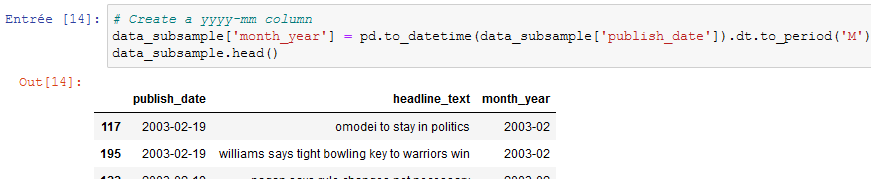
Ah je reconnais cette fonction qui aide a visualiser les premières lignes de donnée



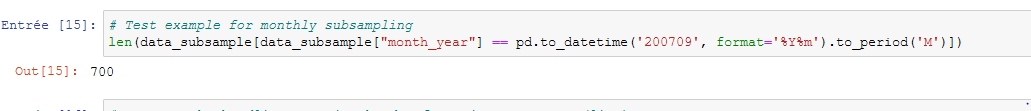
Là je crois qu’on converti le format de la date mais je me demande pourquoi. Est-ce juste une question esthétique (lisibilité) ou pour que le programme fonctionne ?

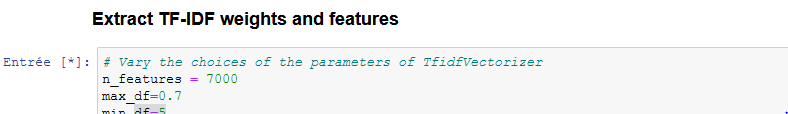


On trie chronologiquement les données

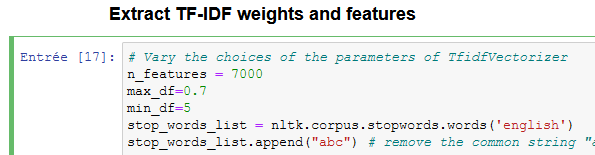


Ok

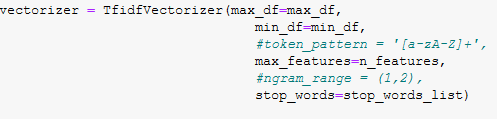




Je me demande ce qu’il extrait vraiment. C’est quoi ces TF-IDF poids et « features ».

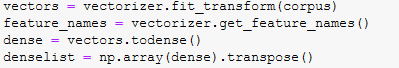


Ça c’est la partie paramétrage

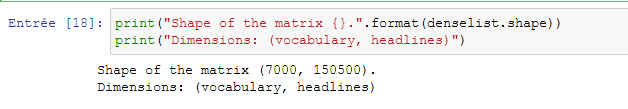


Ils creent ensuite un objet vectorizer de type TfidfVectorizer qui va prendre en entrée tout ce qu’on a créer comme paramètre juste avant.

D’ailleurs pourquoi le df minimum est plus grand que le maximum ? Mystère



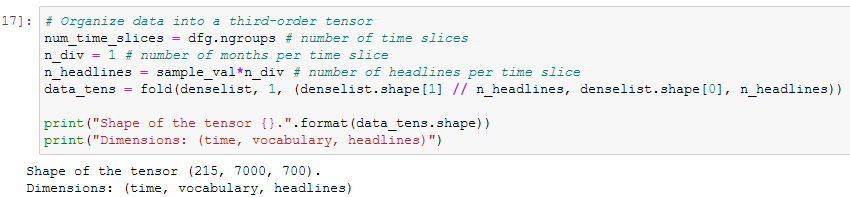
On fait subir des transformations a cet objet jusqu’à ce qu’on obtienne une grosse matrice. La fameuse matrice contenant les mots par document j’imagine :



## 10/06/2022

Bonjour !

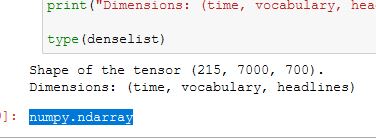
Commençons par le code aujourd’hui. Nous allons voir jusqu’où on va pouvoir l’exécuter. J’espère ensuite pouvoir continuer avec le « tuto » de madame Anupama Garla et pourquoi pas faire notre propre programme de topic modeling.



Mhhm, donc là on fait un tensor ? D’accord.

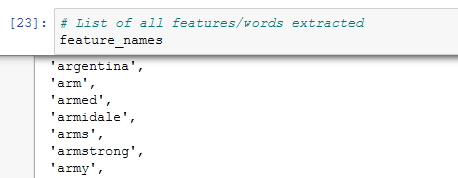
Bon, il faut savoir ce que fait la fonction fold() afin de comprendre comment se construit se tensor.

Il a besoin de cette chose nommée « denselist » qui pour moi est notre matrice a 2 dimension contenant les mots par documents. En suivant cette logique on peut se dire que pour en faire un tensor de dim 3 il faut ajouter la dimension temporelle.



Là j’ai voulu regarder de quel type était denselist. J’aurai aimé faire denselist.head() mais on ne peut pas faire ça avec, ça ne fonctionne pas. Du coup c’est un tableau numpy. Je pense.

Bon ne trainons pas et continuons l’exécution.

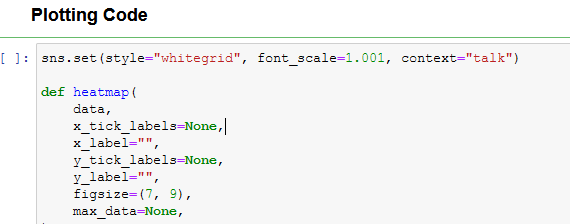


Avec tout ça j’ai oublié ce qu’était un « feature »

Ah d’accord, c’est des mots qu’il extrait. Mais pourquoi faire à cette étape ci ?

Après il faut se rappeler qu’on est dans une partie du programme nommée « extract TF-IDF weights and features ».

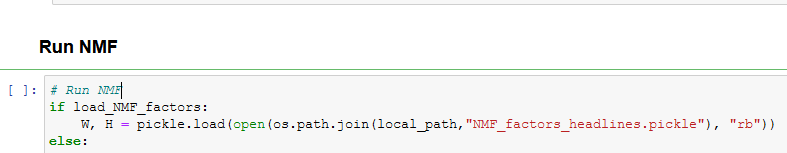
Remarque : Je n’ai beaucoup de recul en ce moment. Peut-être que ça viendra. Je verrai peu à peu comment chacune de ces petites actions s’inscrit dans l’ensemble.



Oh on prépare des graphiques.

On a créé une fonction qui nous aide à générer nos « heatmaps ». On utilise dedans plein de fonctions que je ne connais pas vraiment, je peux essayer de deviner à quoi ils servent en lisant leurs noms.

Passons.

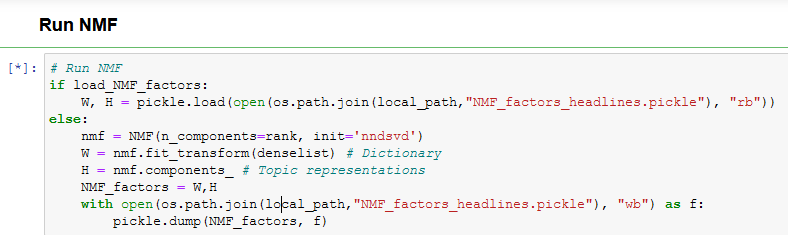


Nous voici à la partie Run NMF. On va donc y faire la NMF. Rappelons qu’on travaille sur des « news headlines » que je traduis par « titres d’article de presse ».

Cette partie du programme se décompose en 5 blocs qu’ils ont pris le temps de nommer respectivement :

* # Run NMF
* # Display topics
* # Topic Keywords representations (intéressant)
* # Get topic distributions for each time slice
* #Visualize topic distributions

Ils contiennent aussi des sous-parties



Il prend tout son temps pour exécuter cette partie. On m’avait dit pourtant que ce programme était plus long à exécuter.

Je propose que l’on fasse autre chose en attendant que mon ordi surpuissant fasse le travail.

Regardons le code d’Anupama Garla (j’adore ce nom) ou bien des vidéos utiles.

Finalement je suis allé voir avec Luan comment les programmes fonctionnent (pas) sur son ordi.

Bon Luan a finalement réussi à faire fonctionner les programmes. Il a dû installer « latex » et le mettre à jour je crois mais je ne suis pas sure.

Je viens de créer un fichier NMF\_tentative\_1.py afin d’essayer d’en faire une à l’aide du tutoriel de Garla.

Je remets le lien ici : <https://towardsdatascience.com/nmf-a-visual-explainer-and-python-implementation-7ecdd73491f8>

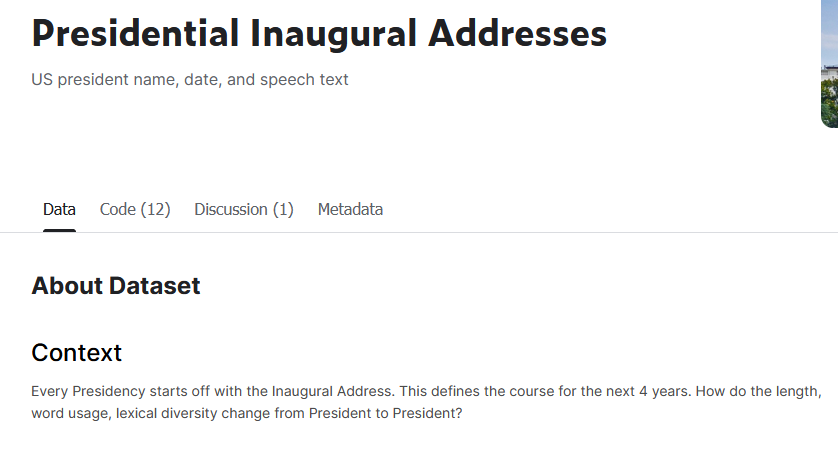
A mesure que j’avance dans le tutoriel de Garla je me rends compte qu’il y a une étape que l’on n’a pas eu dans les programmes précédents. Il s’agit du « text cleaning ». J’ai vu un autre stagiaire qui lui travaillait sur la LDA en faire.

Bon il faut y aller. C’est fini pour aujourd’hui.

## 13/06/2022

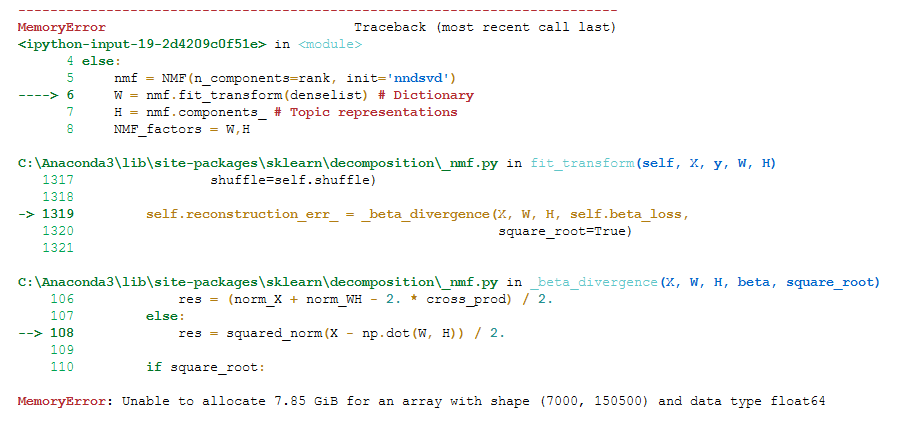
Continuons ce que nous faisions avant ce week-end.

Cela fait un moment que la NMF que j’ai relancé continue de tourner



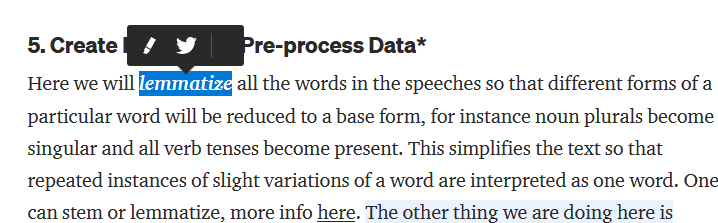
Les données du tutoriel à telecharger

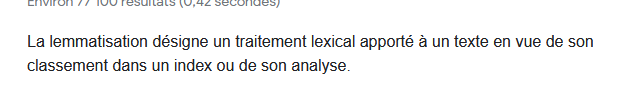
## 14/06/2022

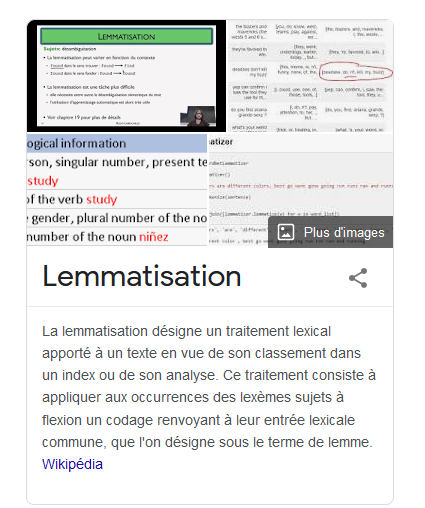


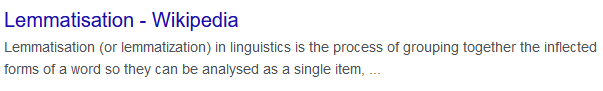
Après quelques heures de fonctionnement il me dit qu’il n’a pas pu allouer la place nécessaire au tableau de données

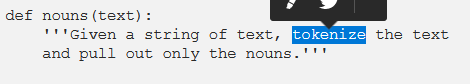
## 16/06/2022

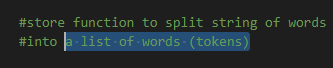


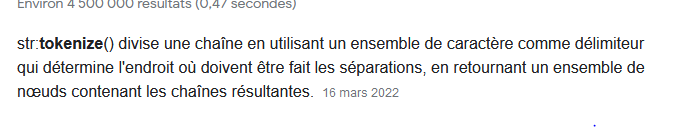


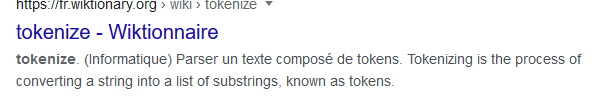












Il faut revoir ce qu’est la SVD afin de mieux comprendre la NMF

On m’a dit que lorsqu’on fait une NMF on fixe alternativement l’une des matrices tandis qu’on transforme l’autre par rapport au premier. Puis on inverse les roles. Je ne sais cependant pas quelle genre de transformation est appliquée.