Projet Python

Lien GitHub : https://github.com/Manu-RMT/M1_SPE_PYTHON

Git clone : git clone https://github.com/Manu-RMT/M1_SPE_PYTHON.git

I-Rappel Problématique/Spécifications/Intro

Aujourd'hui, il existe une multitude de documents que l'on peut retrouver en ligne. L'objectif de ce projet est de développer un outil qui permettrait à des utilisateurs (qui ne sont pas forcément des informaticiens) de comparer des documents. Il faut donc concevoir une interface qui permettrait de rechercher des documents dans un corpus à partir de mots-clefs.

Nous avons choisi de rechercher les documents sur Reddit et ArXiv car nous disposions d'API qui permettent de lancer des recherches de documents à partir de mots-clefs.

Nous avons choisi de créer un fichier csv qui serait le résultat d'une recherche de document sur Reddit et ArXiv. Nous voulions ensuite avoir une interface sur laquelle on pourrait saisir des mots-clefs qui séparerait les documents du csv en deux corpus : le premier contenant au moins un des mots clés et le second avec le reste des documents. Une fois avoir obtenu le corpus correspondant aux mots clefs, on a décidé de l'analyser à travers quelques indicateurs comme la taille du vocabulaire du corpus, les mots avec les plus fortes TF et TFxIDF.

II-Analyse

1.Environnement de travail

Nous avons pris Anaconda comme environnement de travail car il s'agit d'un outil open source qui contient des environnements de développement notamment Spyder qui nous a permis d'écrire notre code. Anaconda contient également des packages que l'on peut installer afin de détenir une multitude de possibilités. Enfin nous l'avions déjà utilisé au préalable, nous savions donc comment en tirer le meilleur parti.

2.Données identifiées dans les spécifications

Nous avons récupéré des documents de ArXiv et Reddit résultant de la requête « football ». On a choisi de commencer par prendre 500 documents Reddit puis de ne conserver que ceux ayant un texte. On a ensuite pris autant de document ArXiv que de documents Reddit que nous avons conservé.

Derrière on a stocké dans un dataframe plusieurs champs :

-Nature (correspond à l'origine du document : Reddit ou ArXiv)

-Texte

-Auteur

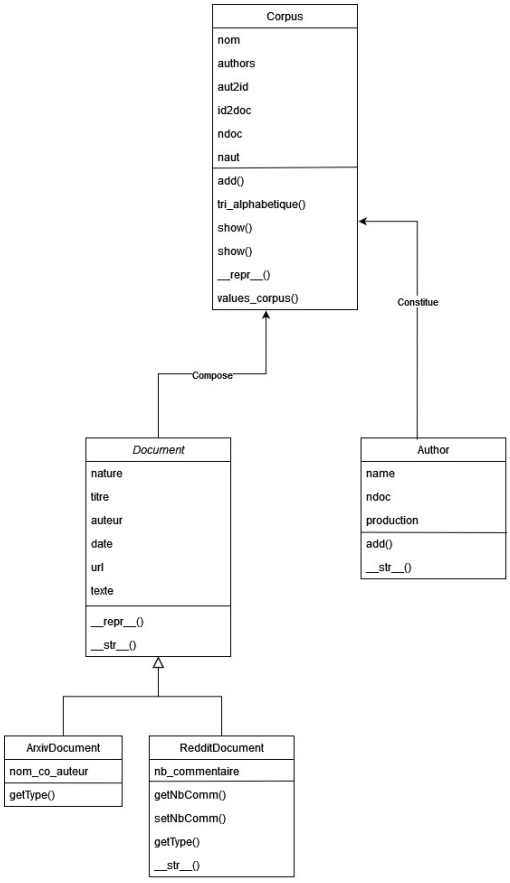
-Titre

-Date

-URL

-Words (Texte tokenisé)

3.Diagramme des classes



III-Conception

1.Partage des tâches

Lorsque nous avons commencé ce projet, nous ne partions pas de 0 car nous avions déjà élaborer des codes qui nous ont servi pour le projet. La première étape était donc de créer un fichier à partir de nos codes qui étaient différents pour partir de la même base.

Ensuite nous avions chacun plus de facilités dans certains domaines. Nous avons donc découpé le travail en tâches que nous nous répartissions selon les acquis de chacun à chaque sprint. A chaque fois, que nous terminions une tâche, nous faisions un point afin d'expliquer le code effectué et de l'incorporer dans le fichier principal. Cela permet à la fois de faire une correction du code par celui qui n'a pas réalisé la tâche et que tout le monde comprennent ce que font les différentes fonctions. Ces points nous ont également permis de choisir les tâches suivantes à réaliser et à se partager.

Une fois la partie code achevée, nous nous sommes réunis pour ajouter les derniers commentaires nécessaires, rédiger le rapport et tester notre code sur un nouvel environnement en téléchargeant les packages avec notre fichier « requirements ».

2.Problèmes rencontrés

- Création du csv (parenthèses, textes vides)

La sauvegarde des documents provenant des API dans un CSV nous a aussi pris un peu de temps à réaliser. Le problème principal était que nous avions créé les documents puis les rangions dans le corpus avec « variable = document », on ne récupérait alors que la sortie de __str__ au lieu des différents champs de la classe document. Nous avons donc modifié la sortie du __str__ afin qu'elle soit composée des champs de la classe Document avec des « ; » entre chacun des champs.

- Tri de dictionnaire par valeur et non pas par clé

N'ayant pas souvent fait l'usage de dictionnaire nous avons eu du mal à trier les dictionnaires par valeur et non par clé. Cela étant nécessaire dans notre code afin de classer les mots par TF ou TFxIDF afin de ne montrer que ceux étant les plus (ou moins) importants.

- Insertion de plusieurs mots-clefs

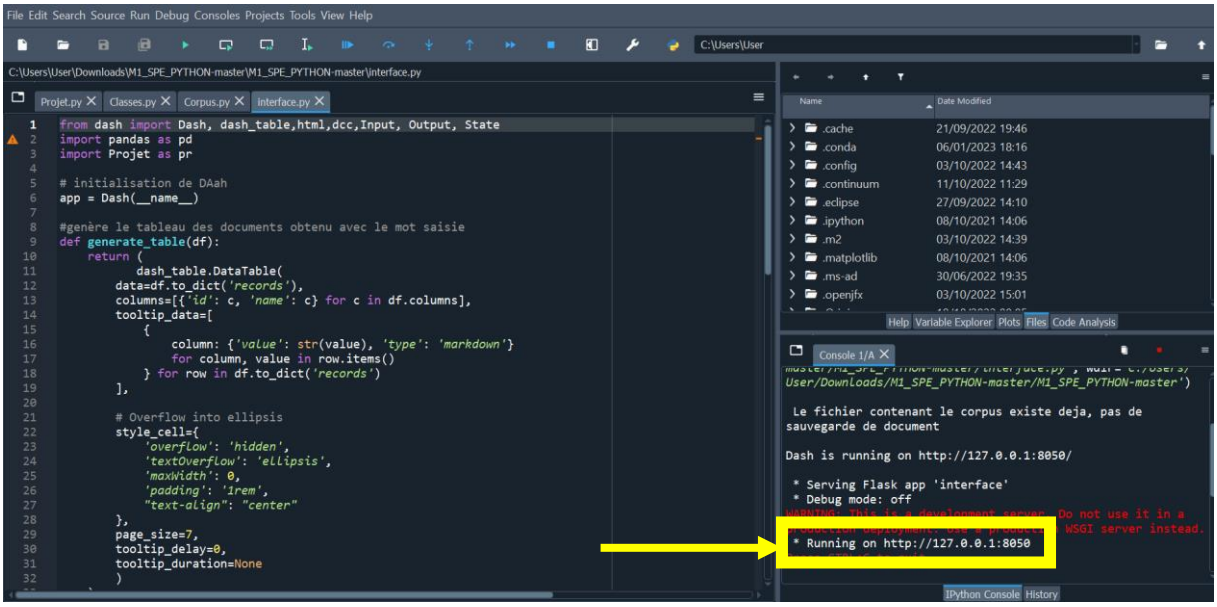
Nous avons rencontré un autre problème, au début nous n'avions configuré la saisie de mots-clefs dans l'interface que pour un unique mot. Nous nous sommes ensuite dit que la saisie de plusieurs mots pouvait être intéressante. Nous avons donc dû passer d'une variable simple à une liste que nous avons découpés à l'aide de split. Après l'avoir découpé, il fallait transformer la liste ['mot1','mot2'] par une chaine de caractère « mot1 ou mot2 » que nous avons fait avec la fonction join. Enfin nous avons aussi dû régler une autre erreur. Lorsque l'on rajoutait un espace sans mot derrière lui dans la saisie de mot clefs, on avait un « ou » de trop dans notre chaîne de caractère. Il a donc fallu procéder à un filtre pour supprimer cela.

- Peu d'exemples de code dash sur internet

Dash semble ne pas être très utilisé dans le monde professionnel. En effet il était difficile de trouver de l'aide/des exemples de code dash. On a donc dû passer beaucoup de temps à comprendre le fonctionnement de dash.

3.Exemple commenté d'utilisation du programme

Afin de lancer le programme, il faut lancer le code du fichier « interface.py ». On obtient alors un lien qu'il faudra copier et ouvrir sur un navigateur internet.



On a alors cette fenêtre qui nous demande de rentrer un mot :

On peut donc rentrer des mots-clefs séparer par des espaces et cliquer sur le bouton « Submit ».

Si nous tapons « France » et « FIFA » puis cliquons sur « Submit », notre interface ressemblera à ceci :

Exploration de documents de football

Barre de recherche France FIFA Rechercher

34 documents contiennent le mot france ou fifa et constituent le corpus value

750 documents ne contiennent pas le mot france ou fifa et constituent le corpus novalue

Taille du vocabulaire des documents contenant un des mots-clés : 1895 mots

mots avec le plus grand tf : ['world', 'teams', 'cup', 'th', 'like', 'football', 'france', 'players', 'team', 'messi', 'would', 'fifa', 'position', 'think', 'average', 'nations', 'also', 'argentina', 'one', 'final']

mots avec le plus grand tfxidf : ['download', 'full', 'length', 'fifa', 'every', 'week', 'try', 'make', 'new', 'simulation', 'cupso', 'far', 'simulated', 'following', 'cups', 'worst', 'nations', 'confederation', 'unfortunately', 'stupid']

Tableau du corpus value

Nature	Titre	Auteur	Date	URL	Texte
Reddit	Query related to footba…	kuxwar	2023/01/05	https://www.reddit.com/…	Where to download full …
Reddit	A World Cup - but only …	1seven6	2023/01/04	https://www.reddit.com/…	Every week I try to mak…
Reddit	Argentina revolutionize…	ForlanMatador	2023/01/05	https://www.reddit.com/…	Scaloni is a genius he …
Reddit	Chelsea - Benfica Deal …	Ppais89	2023/01/04	https://www.reddit.com/…	Last Time I publish som…

On a donc le nombre de documents qui comportent au moins « France » ou « FIFA » et le nombre de documents qui ne comportent aucun de ces mots.

Ensuite on a les mots avec les plus grand TF puis les mots avec les plus grands TFxIDF parmi les documents du corpus « value » (qui est celui qui comporte au moins un des mots-clefs et qui est affiché plus bas).

En survolant le tableau, on peut lire le texte et en cliquant sur une cellule on peut copier l'ensemble de son contenu.

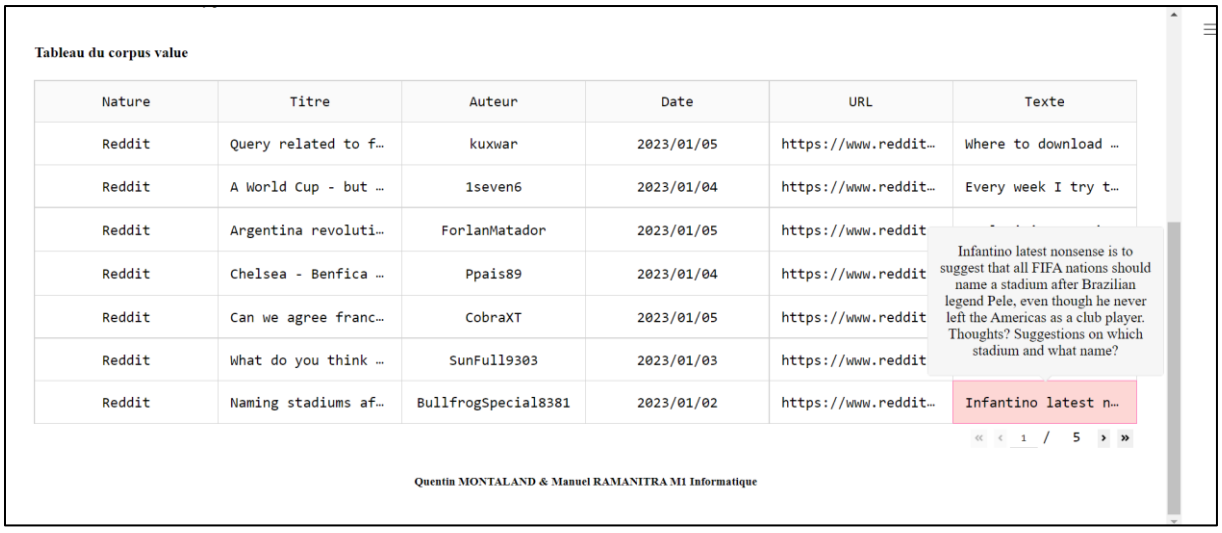


Tableau du corpus value

Nature	Titre	Auteur	Date	URL	Texte
Reddit	Query related to f…	kuxwar	2023/01/05	https://www.reddit…	Where to download …
Reddit	A World Cup - but …	1seven6	2023/01/04	https://www.reddit…	Every week I try t…
Reddit	Argentina revoluti…	ForlanMatador	2023/01/05	https://www.reddit	
Reddit	Chelsea - Benfica …	Ppais89	2023/01/04	https://www.reddit	
Reddit	Can we agree franc…	CobraXT	2023/01/05	https://www.reddit	
Reddit	What do you think …	SunFull9303	2023/01/03	https://www.reddit	
Reddit	Naming stadiums af…	BullfrogSpecial8381	2023/01/02	https://www.reddit…	Infantino latest n…

Infantino latest nonsense is to suggest that all FIFA nations should name a stadium after Brazilian legend Pele, even though he never left the Americas as a club player. Thoughts? Suggestions on which stadium and what name?

« ‹ 1 / 5 › »

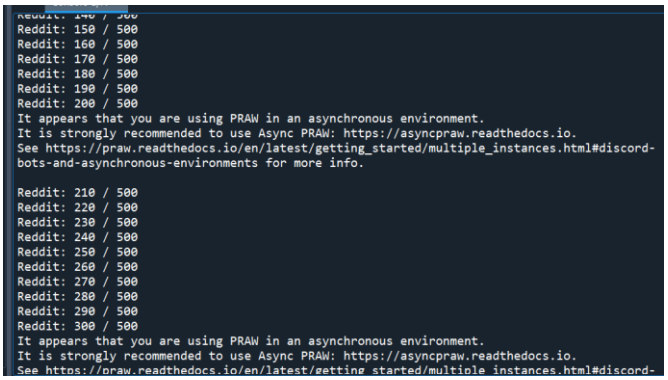
Quentin MONTALAND & Manuel RAMANITRA M1 Informatique

IV-Validation

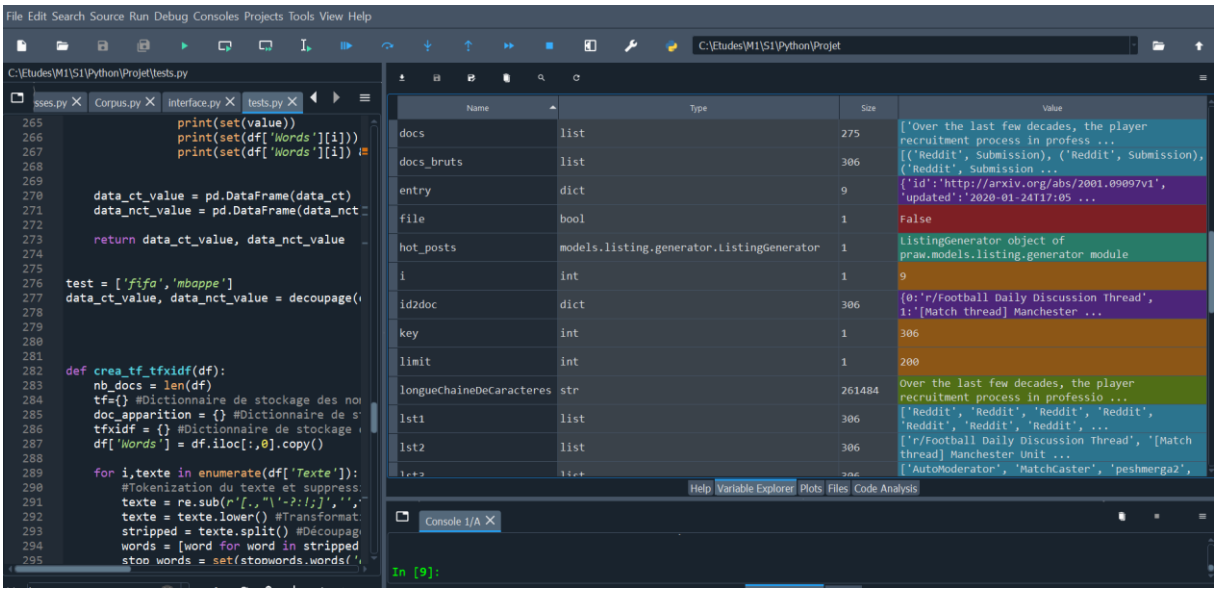
1.Tests unitaires

Le projet étant constitué de plusieurs classes et fonctions, il nous a été important de réaliser des tests unitaires notamment pour savoir où était le problème lorsque nous n'avions pas le résultat espéré.

Ces tests sont passés à la fois par des « print » au sein de fonction ou de boucles afin de savoir combien de fois les fonctions/boucles étaient lancés.



On a aussi créé un nouveau fichier dans lequel on retirait le code des fonctions ce qui nous permettait de pouvoir observer les valeurs ainsi que les tailles des variables/listes/dictionnaires/classes créés dans l'onglet « Variable Explorer » de Spyder.



2.Tests globaux

Nous avons effectué plusieurs tests globaux lors de ce projet.

Nous avons déjà évoqué que nous avons lancer notre code sur un nouvel environnement sur lequel nous avons utilisé notre fichier « requirements » afin de télécharger les packages nécessaires.

Sur notre interface, nous avons essayé de rentrer plusieurs mots-clefs, de rentrer des mots clefs avec des majuscules. Ces tests nous ont été la cause de nouveaux développements par la suite.

V-Conclusion et perspectives/Maintenance

Lors de ce projet, nous avons réussi à créer un fichier csv à partir de requêtes d'API, puis permettre la recherche de certains termes sur une interface en accompagnant cette recherche de quelques informations (mots les plus importants avec les TF et TFxIDF ainsi que le nombre de documents correspondants à la recherche et le nombre de mots différents dans le corpus). Nous avons donc augmenté nos compétences techniques sur Python ainsi que nos compétences « projet » puisque nous sommes bien arrivés à nous départager le travail en prenant en compte les capacités de chacun ainsi qu'en arrivant bien à regrouper nos tâches.

Pour les perspectives, nous avons eu quelques idées d'améliorations :

- Nous pensons aussi que nous aurions pu améliorer notre application en préchargeant des données dans plusieurs csv étant chacun la conclusion de requête différentes sur Reddit et ArXiv. L'utilisateur aurait alors dû choisir un des fichiers csv pour ensuite effectuer une recherche avec ses mots-clefs. La création de plusieurs csv n'aurait pas été un problème, il aurait ensuite fallu créer une liste sur l'interface afin que l'utilisateur puisse faire son choix.

- Nous avons également pensé qu'il serait bien de pouvoir filtrer le tableau affiché par auteur ainsi que de pouvoir le trié par auteur/date/nature.

- Nous aurions également pu faire une opposition entre le corpus contenant au moins un mot clé et celui n'en contenant pas. C'est ce que nous avons essayé de faire en comparant les valeurs des TF ou TFxIDF des mots ressortant le plus. Le problème était que le calcul des TF et TFxIDF prenait beaucoup de temps pour le deuxième corpus (car il était composé de bien plus de documents). Nous avons donc choisi de nous intéresser seulement au corpus comprenant les mots-clefs