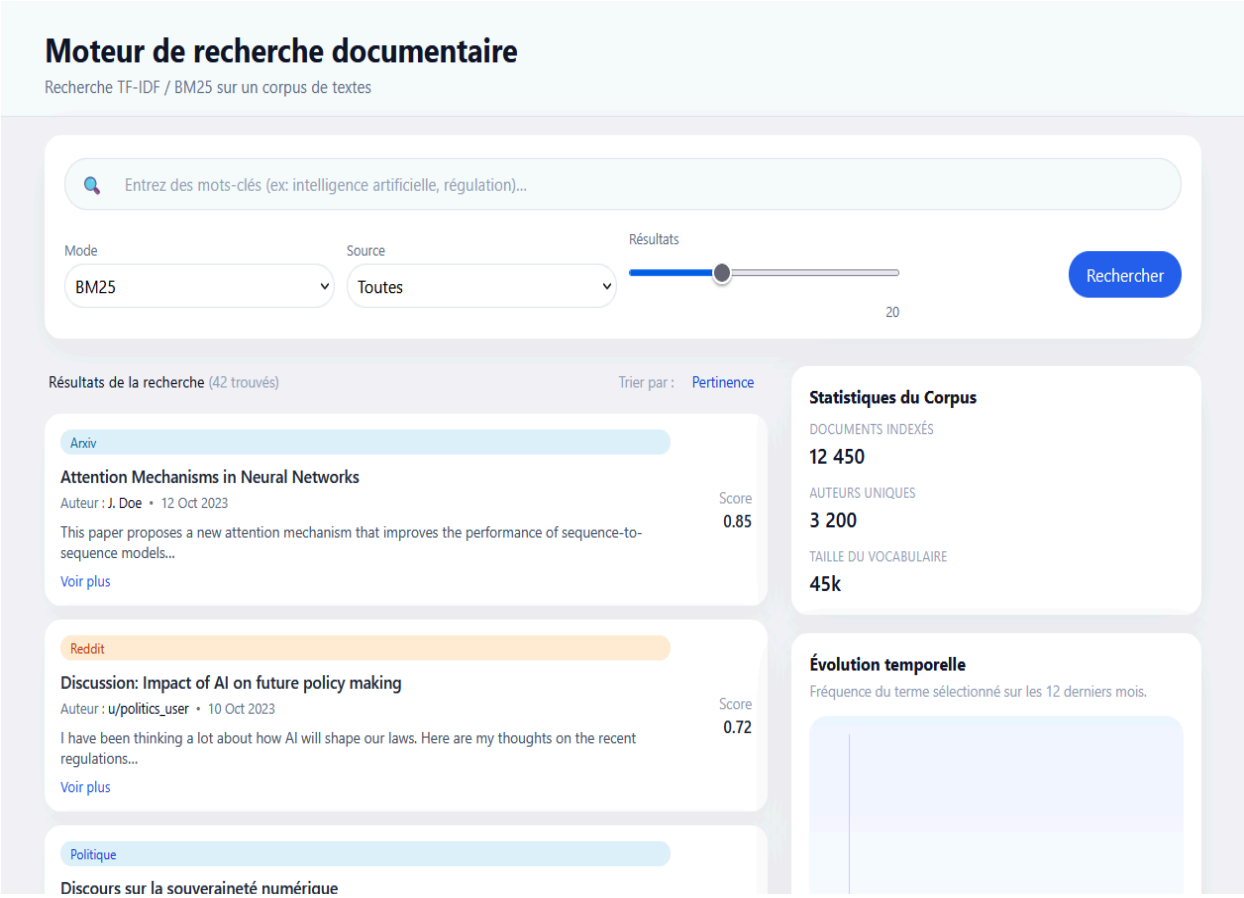
Projet Python - Rapport

MOTEUR DE RECHERCHE

Cybersécurité



par Selma IMANSOUREN, Ali SAÏDI et Andréa LYONNET, le 18 Décembre 2025

lien du dépôt github : https://github.com/hushpuppyy/Projet-python

Introduction

Ce projet s'inscrit dans le cadre des TD n°3 à TD n°10 du cours de Programmation Python.
L'objectif global était de réaliser un **moteur de recherche textuel** complet, en suivant les différentes étapes classiques d'un cycle de développement logiciel :

1. spécifications
2. analyse,
3. conception,
4. codage,
5. intégration,
6. validation,
7. ajout d'une interface utilisateur.

Le projet a consisté à manipuler des corpus textuels réels, à les structurer en objets Python, à construire des représentations vectorielles, à calculer des similarités entre documents, et enfin à développer une petite interface utilisateur basée sur **ipywidgets** intégrée dans un notebook Jupyter.

Le rendu final correspond à la version final = moteur de recherche complet + interface graphique (page html dans notre cas)

1. Charger un corpus textuel stocké dans un fichier.
2. Structurer chaque élément du corpus sous forme d'objets Python.
3. Construire automatiquement un index inversé et une matrice TF-IDF.
4. Permettre des requêtes textuelles et de renvoyer les documents les plus pertinents.
5. Fournir un résumé lisible des résultats (titre, auteur, extrait, score).
6. Proposer une interface utilisateur simple permettant :
    - de saisir une requête,
    - de choisir un nombre maximal de résultats,
    - d'afficher en direct la sortie dans un widget Jupyter.

Environnement de travail

Langage : Python 3.13

IDE : VSCode

Kernel : environnement virtuel .venv, Python

Outils complémentaires :

- Jupyter Notebooks
- ipywidgets
- matplotlib
- tqdm

Données : fichiers discours_US.tsv, Reddit, Arxiv

Chaque entrée du corpus comporte les champs suivants :

- *speaker* : auteur du discours
- *text* : texte intégral du discours
- *date* : date du discours
- *descr* : description / titre
- *link* : URL source

Classe Document

- titre, auteur, date, url, texte, type

Classe Corpus

- nom du corpus, id2doc : dictionnaire {id → Document}, authors : dictionnaire {auteur → liste de Documents}
- méthodes :
  - add_document()
  - search() (pour motifs simples, TD6)

- concorde()
- stats()
- nettoyage de texte

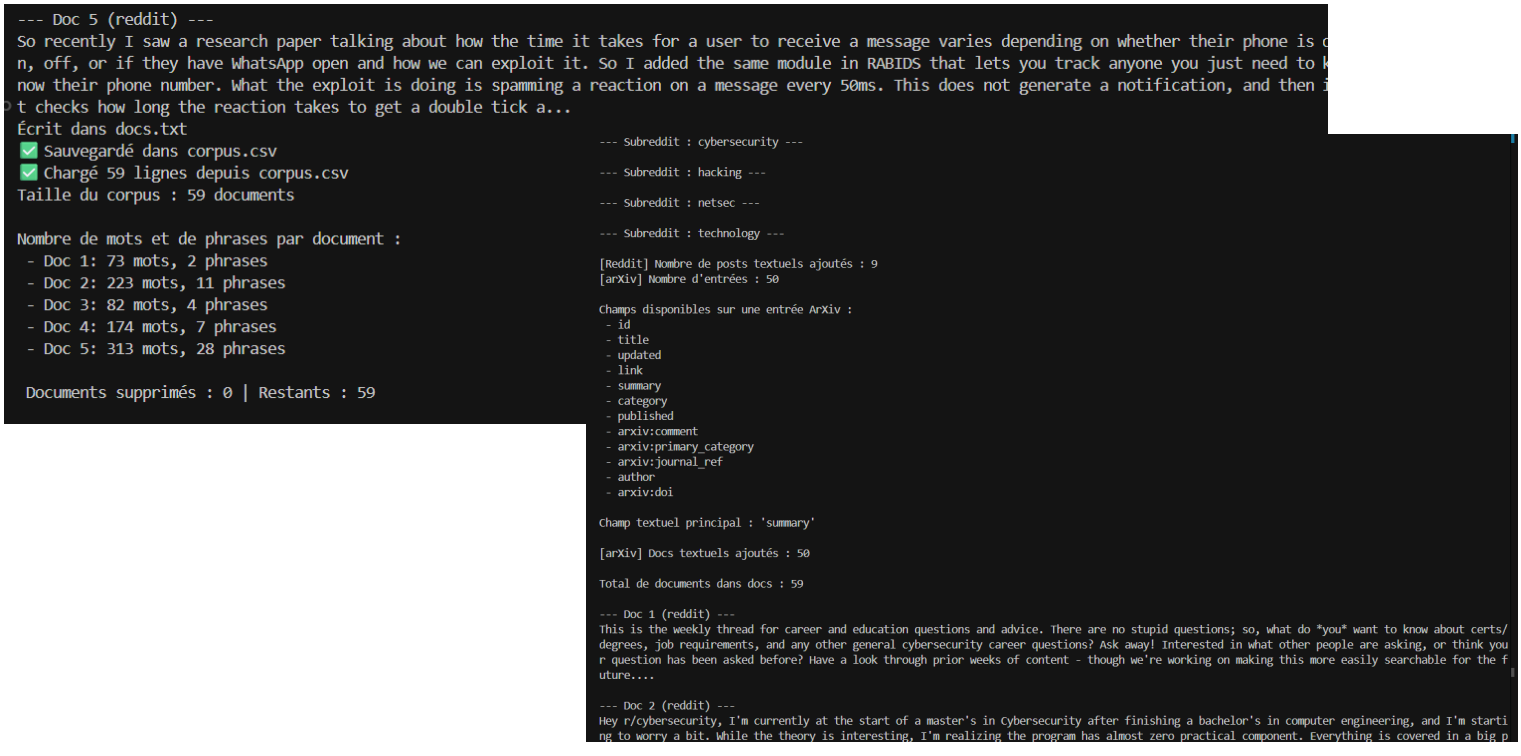
Classe SearchEngine

- construire le vocabulaire, construire la matrice TF et TF-IDF, calculer les similarités cosinus, retourner un DataFrame trié par score, gérer la vectorisation des requêtes
- **Corpus → Document** : relation de composition (un corpus contient des documents).
- **Corpus → SearchEngine** : relation d'utilisation (le moteur lit les documents).
- **SearchEngine → Document** : le moteur utilise les textes des documents pour construire les vecteurs.
- **Interface → SearchEngine** : l'interface appelle la fonction search.

Présentation des TD

TD3 – Acquisition de données

L'objectif de ce TD est de collecter automatiquement des documents depuis Reddit et ArXiv afin de constituer un corpus.



TD4 – Construction du vocabulaire et statistiques

Ce TD consiste à construire le vocabulaire du corpus et à calculer les statistiques associées comme la fréquence des mots (TF) et le nombre de documents (DF).

```
– CyberSec: top 7 par date –
[2] 2025-12-12 14:50:18 | MSc in Cybersecurity is teaching me nothing practical, any advice? | TheGroovyKiwi
[3] 2025-12-12 13:42:01 | Negotiating with cybersecurity vendors | greenclosettree
[4] 2025-12-11 15:51:11 | Choice between SOC analyst and Sysadmin with Security responsibilities | Auno94
[9] 2025-12-10 08:39:28 | Free Honey Tokens for Breach Detection - No Signup | radkawar
[7] 2025-12-08 10:24:56 | FUD Crypters in 2025? | No_Law3758
[1] 2025-12-08 01:00:35 | Mentorship Monday - Post All Career, Education and Job questions here! | AutoModerator
[6] 2025-12-07 23:26:49 | Should I learn the CCNA or network+ curriculum to learn the computer networking  | SkibidiRizzSus

--- Top 7 par titre ---
– CyberSec: top 7 par titre –
[20] A Global Analysis of Cyber Threats to the Energy Sector: "Currents of Conflict"  | Gustavo Sánchez
[5] A WhatsApp Exploit that let you track anyone | Impossible_Process99
[16] Assembling a Cyber Range to Evaluate Artificial Intelligence / Machine Learning  | Jeffrey A. Nichols
[10] Assessing Cyber-Physical Security in Industrial Control Systems | Martín Barrère
[17] Challenges in Digital Twin Development for Cyber-Physical Production Systems | Heejong Park
[4] Choice between SOC analyst and Sysadmin with Security responsibilities | Auno94
[23] Cyber Pirates Ahoy! An Analysis of Cybersecurity Challenges in the Shipping Indu | George Grispos

=== Corpus rechargé depuis corpus.tsv ===
Corpus('CyberSec', ndoc=29, naut=27)
– CyberSec: top 3 par titre –
[20] A Global Analysis of Cyber Threats to the Energy Sector: "Currents of Conflict"  | Gustavo Sánchez
[5] A WhatsApp Exploit that let you track anyone | Impossible_Process99
[16] Assembling a Cyber Range to Evaluate Artificial Intelligence / Machine Learning  | Jeffrey A. Nichols

Auteur pour statistiques : George Grispos

Auteur : George Grispos
Docs : 1
Taille moyenne : 784.0 caractères
Titres :
 - (23) Cyber Pirates Ahoy! An Analysis of Cybersecurity Challenges in the Shipping Indu

Singleton Corpus ? : True
```

TD5 – Représentation vectorielle des documents

Dans ce TD, les documents sont transformés en vecteurs numériques à l'aide des matrices TF et TF-IDF afin de permettre leur comparaison.

```
[PARTIE 1] Motif (regex) à chercher dans le corpus : cyber

Nombre d'occurrences trouvées pour 'cyber' : 106

Quelques extraits :
... job requirements, and any other general cybersecurity career questions? Ask away!

In
------------------------------------------------------------
... easily searchable for the future. Hey r/cybersecurity,

I'm currently at the start of
------------------------------------------------------------
... currently at the start of a master's in Cybersecurity after finishing a bachelor's in
------------------------------------------------------------
... 200, TCM PSAA).

**TL;DR**: Master's in Cybersecurity is giving me theory but no prac
------------------------------------------------------------
... prove our deals / contracts for several cyber security solutions I’m managing. Is the
------------------------------------------------------------

Aperçu du concordancier :
                              contexte_gauche motif_trouve                              contexte_droit
0    job requirements, and any other general         cyber  security career questions? Ask away!\n\nIn
1    easily searchable for the future. Hey r/        cyber  security,\n\nI'm currently at the start of
2    currently at the start of a master's in         Cyber    security after finishing a bachelor's in
3  200, TCM PSAA).\n\n**TL;DR**: Master's in         Cyber    security is giving me theory but no prac
4    prove our deals / contracts for several         cyber     security solutions I’m managing. Is the

Concordancier sauvegardé dans 'concordancier.tsv'.
```

TD6 – Mesure de similarité

Ce TD introduit les mesures de similarité (cosinus) pour comparer des documents entre eux ou avec une requête utilisateur.

```
Aperçu du tableau freq (tf/df) :
          word   tf  df
1462       the  231  28
1117       and  178  28
1458        of  151  27
604         to  131  27
1065         a  103  27
690         in  101  27
1093     cyber   85  23
948        for   64  24
1135        is   63  24
1591        on   58  22
1422  security   57  16
1524        we   56  17
1435      that   51  22
952       this   49  23
1110       are   43  22
1415         i   41  11
368         or   28  15
552         be   27  14
692         an   27  15
237       with   27  17

Tableau 'freq' sauvegardé dans 'freq.tsv'.
```

```
[PARTIE 2] Combien de mots les plus fréquents afficher ? (n) : 5

=== STATISTIQUES TEXTE ===
Nombre total de mots distincts : 1617

Top 5 mots les plus fréquents :
     word   tf  df
1462  the  231  28
1117  and  178  28
1458   of  151  27
604    to  131  27
1065    a  103  27

Aperçu du tableau freq (tf/df) :
          word   tf  df
1462       the  231  28
1117       and  178  28
1458        of  151  27
604         to  131  27
1065         a  103  27
690         in  101  27
1093     cyber   85  23
948        for   64  24
Nombre total de mots distincts : 1617

Top 5 mots les plus fréquents :
     word   tf  df
1462  the  231  28
1117  and  178  28
1458   of  151  27
604    to  131  27
1065    a  103  27

Aperçu du tableau freq (tf/df) :
```

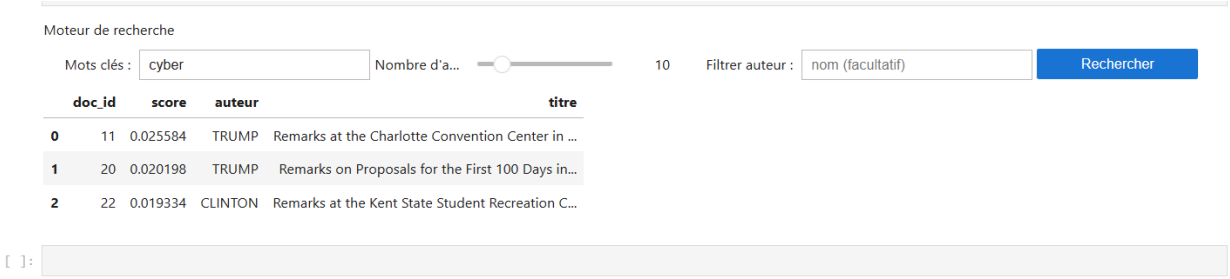
TD7 – Moteur de recherche

Ce TD met en œuvre un moteur de recherche complet permettant à l'utilisateur de saisir des mots-clés et d'obtenir les documents les plus pertinents.

```
Corpus chargé : Corpus('CyberSec', ndoc=29, naut=27)

=== Construction de l'index (TD7) ===
- Nombre de documents : 29
- Taille du vocabulaire : 1617
=== Index construit ===

Entrez quelques mots-clés (ou 'quit' pour arrêter) : cyber attack ai
Combien de documents à afficher ? (n) : 3

=== Meilleurs résultats ===
 doc_id    score  type                                                                                titre            auteur
     16 0.188200 arxiv Assembling a Cyber Range to Evaluate Artificial Intelligence / Machine Learning (AI/ML) Security Tools Jeffrey A. Nichols
     12 0.070338 arxiv                          Security Modelling for Cyber-Physical Systems: A Systematic Literature Review      Shaofei Huang
     28 0.041614 arxiv                                    Fundamental Concepts of Cyber Resilience: Introduction and Overview        Igor Linkov

Résultats détaillés sauvegardés dans resultats_recherche.tsv

Entrez quelques mots-clés (ou 'quit' pour arrêter) :
```

TD8 – Visualisation et exploration

Ce TD ajoute une dimension exploratoire avec des visualisations (évolution temporelle des mots, graphiques, widgets interactifs).

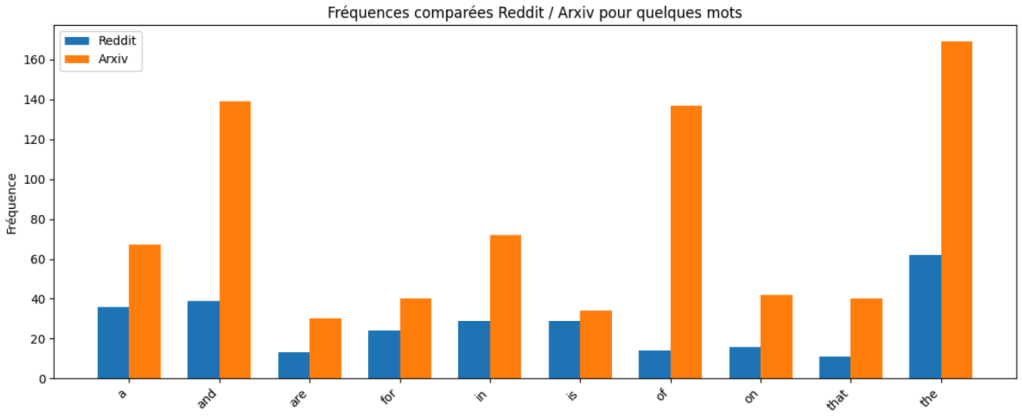


	doc_id	score	auteur	titre
0	11	0.025584	TRUMP	Remarks at the Charlotte Convention Center in ...
1	20	0.020198	TRUMP	Remarks on Proposals for the First 100 Days in...
2	22	0.019334	CLINTON	Remarks at the Kent State Student Recreation C...

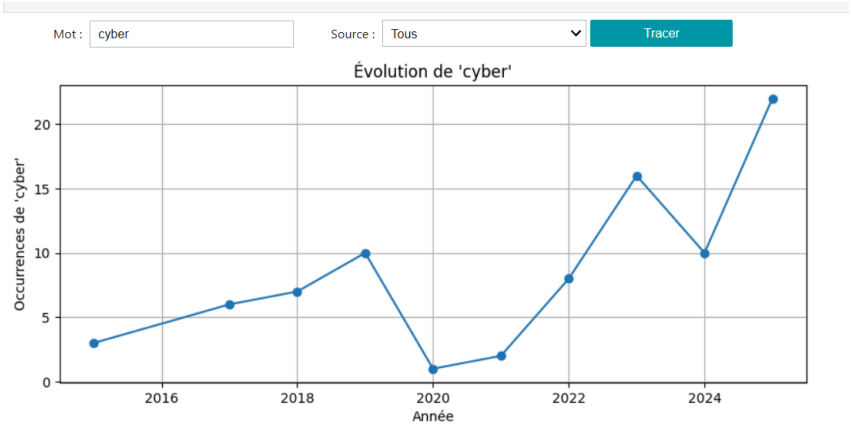
[ ]:

TD9–TD10 — Analyse comparée Reddit / Arxiv + mini-interface

Ce TD charge un **corpus mixte (Reddit + Arxiv + Discours)**, puis réalise une **analyse statistique du vocabulaire** en comparant les fréquences de mots selon la source.
Il commence par vérifier la présence de **ipywidgets** puis :

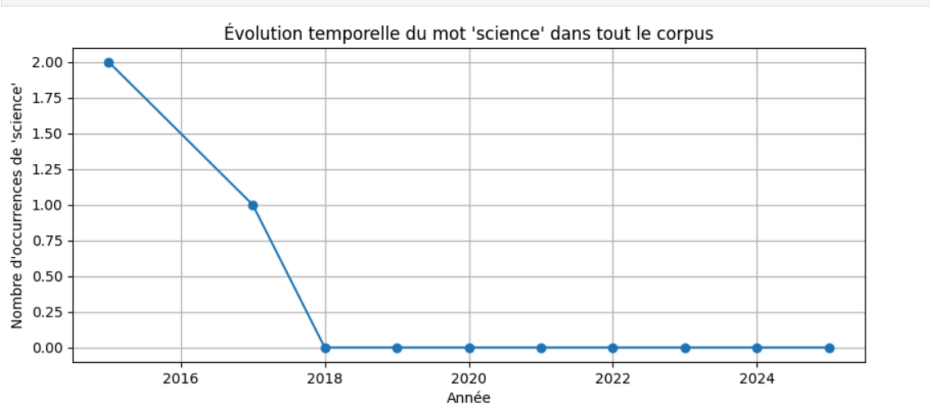


- Charge le corpus et affiche le nombre de documents.
- Calcule la répartition des documents par source



- Définit une fonction top_words_by_type() pour extraire les **mots les plus fréquents** selon le type de document.

- Construit un **tableau comparatif** des top mots Reddit/Arxiv
- Produit un **graphique en barres** pour visualiser la différence de fréquences sur quelques mots communs.



- Définit word_frequency_by_year() pour suivre l'**évolution d'un mot par année**, éventuellement filtrée par source.
- Affiche un **graphique temporel** montrant la fréquence du mot choisi au fil des années.

Analyse

Traitement des textes

Chaque discours est découpé en phrases à l'aide d'une expression régulière simple (re.split(r"[\.?!]", texte)), afin d'obtenir un ensemble de documents courts et indépendants.

Un nettoyage minimal du texte est appliqué :

- mise en minuscules,
- retrait de la ponctuation,
- retrait des chiffres,
- normalisation des espaces.

Construction de l'index et représentation vectorielle

1. Construction du vocabulaire

Pour chaque document :

- découpe en tokens,

- ajout dans un dictionnaire vocab = {mot → colonne},
- parallèlement, liste id2word = [mot1, mot2, ...].

2. Construction de la matrice TF

On construit à la main une matrice creuse CSR en stockant :

- rows : indices de documents
- cols : indices de mots
- data : fréquences TF

3. Calcul de l'IDF

Pour chaque mot :

- df = nombre de documents contenant ce mot
- idf = log(N / df)

4. Matrice TF-IDF

mat_tfidf = mat_tf * idf

5. Normalisation des vecteurs documents

doc_norms = sqrt(sum(mat_tfidf^2))

Permet le calcul rapide de la similarité cosinus.

Recherche

Pour une requête :

1. Le texte est nettoyé et vectorisé selon le vocabulaire.
2. On applique TF-IDF sur la requête selon les IDF calculés.
3. On calcule les scores cosinus entre la requête et chaque document.
4. On trie les résultats et on renvoie les top *n* documents.

Problèmes rencontrés et solutions

Problème	**Solution**
Le fichier CSV des discours était mal lu.	Utilisation de sep="\t" (TSV) et on_bad_lines="skip".
Widgets Jupyter ne s'affichaient pas.	Installation du renderer manquant (Jupyter Widget Renderer). Installation de toutes les dependance pour jupyter notebook
Similarité renvoyait des scores nuls.	Ajout de la normalisation et vérification Token → vocab.
Texte très long entraînant lenteur.	Découpage en phrases + utilisation efficace de matrices creuses.
Barres de progression nécessaires.	Utilisation de tqdm dans la boucle de scoring. Installation
Connexion à l'API Reddit impossible (erreur d'authentification / rate limit).	Création d'une application Reddit (client_id, client_secret, user_agent) et utilisation de l'API via praw avec gestion des limites de requêtes.
Extensions Jupyter non reconnues dans VS Code.	Installation et sélection du bon kernel Python + extensions VS Code (Jupyter, Python, Jupyter Notebook Renderers) et redémarrage du kernel. changement de VsCode a Jupyter notebook
Récupération des données ArXiv incomplète ou lente.	Utilisation de la librairie arxiv avec pagination des résultats et limitation du nombre d'articles par requête pour éviter les timeouts.

Conception

Organisation du code

Le code suit une structure orientée objet :

- document.py → définition de la classe Document
- corpus.py → structure des textes, recherche simple, statistiques
- search_engine.py → vectorisation et moteur TF-IDF
- notebook TD 8.ipynb → interface Jupyter
- notebook TD 9-10.ipynb → interface Jupyter -> transformé en page html

Répartition des tâches

ALI -> TD 8 a 10 + Interface jupyter notebook

SELMA -> Rapport + partie visuel

ANDREA -> TD 3 a 7 + Interface HTML

Choix techniques

- Matrice csr_matrix (scipy) pour efficacité mémoire.
- Nettoyage minimal pour éviter perte d'information.
- Découpage en phrases pour augmenter la granularité.
- TF-IDF classique pour pondération.
- Similarité cosinus pour pertinence.
- Widgets ipywidgets pour une interface légère .

Interface utilisateur

L'interface reproduit le modèle demandé dans le support du cours.
Elle utilise les widgets suivants :

- Label → titre + libellés
- Text → saisie de la requête
- IntSlider → choix du nombre de documents
- Button → déclenchement de la recherche
- Output → zone d'affichage des résultats
- Dispositions : HBox, VBox

La fonction clique_bouton assure la collecte des valeurs, l'appel au moteur et l'affichage.

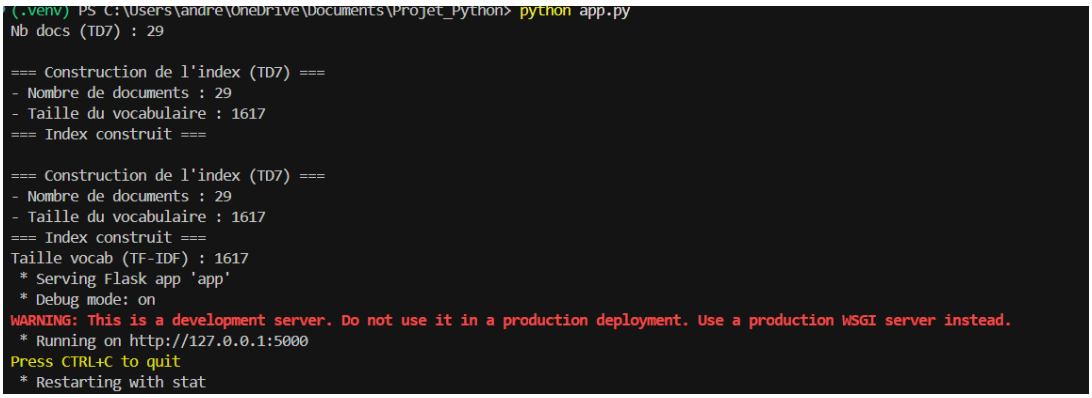
L'ensemble permet une interaction fluide directement dans le notebook.

Page HTML

Page HTML (moteur de recherche cyber)

Comment lancer

- Ouvrir un terminal dans le dossier du projet (là où se trouve `app.py`).

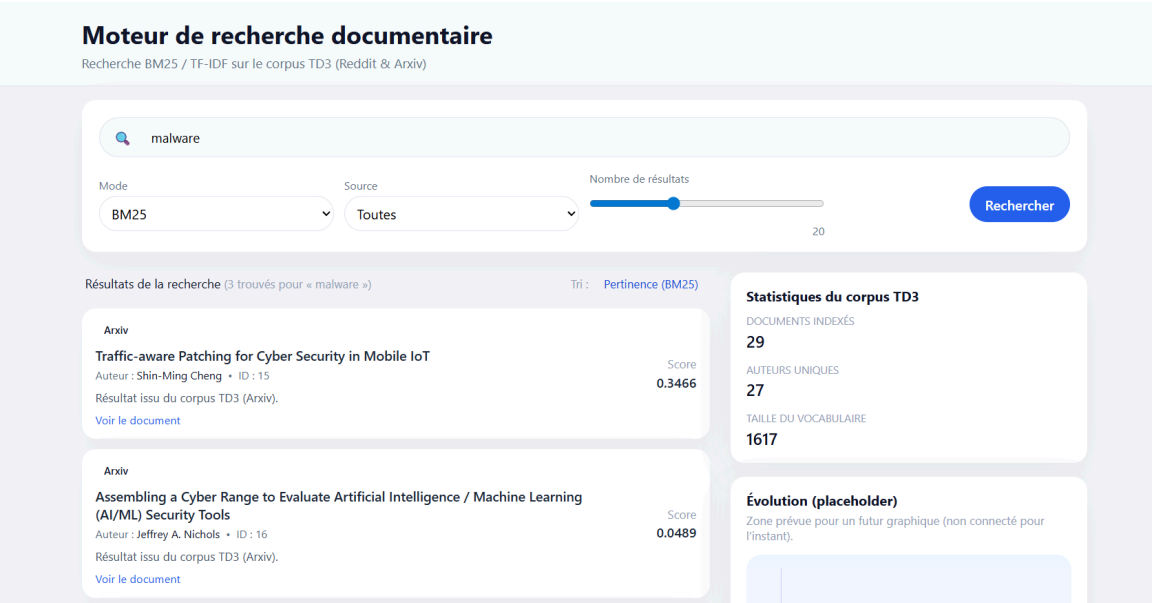


- Activer l'environnement Python si besoin (`.venv`).
- Installer Flask si nécessaire : `pip install flask`
- Lancer le serveur :
    - `python app.py`
- Ouvrir le navigateur sur l'adresse affichée (`http://127.0.0.1:5000/`).

Fonctionnement

Cette page HTML représente une **interface de moteur de recherche orienté cybersécurité** (barre de recherche + affichage des résultats). Elle ne fait pas la recherche "toute seule" : elle s'appuie sur le travail du **TD7**, notamment :

- le **Corpus** (collection de documents : posts Reddit / articles Arxiv, etc.)
- le **SearchEngine** (construction du vocabulaire, matrice TF/TF-IDF, calcul de similarité et classement des documents)



Le **petit serveur Flask** sert d'intermédiaire :

- Flask affiche la page (`route /`)
- Quand l'utilisateur saisit une requête et valide, la page envoie la requête au serveur

- Le serveur appelle `SearchEngine.search(...)` sur le **corpus du TD7**
- Flask renvoie ensuite les **résultats triés** (titres, extraits, score, lien...) vers la page HTML, qui les affiche.

Conclusion

Ce projet nous a permis de mettre en pratique énormément de notions en **Python**, aussi bien sur le plan algorithmique que sur la structuration du code. Chaque TD constituait une étape clé, les précédents servant de fondation aux suivants, ce qui a permis une progression logique jusqu'à la réalisation d'un **moteur de recherche complet**.

Plus précisément, ce projet a permis :

- d'apprendre à **structurer et exploiter un corpus textuel**,
- d'utiliser **Python sur des données réelles**,
- de manipuler des **représentations vectorielles avancées** (TF, TF-IDF),
- de **concevoir un moteur de recherche fonctionnel**,
- et d'intégrer une **interface interactive pratique** .

La liberté laissée dans les choix d'implémentation a favorisé l'autonomie, l'expérimentation et une meilleure compréhension des mécanismes internes, sans recourir à des bibliothèques clé .

Enfin, nous sommes **fiers du rendu final**, qui combine analyse de données, moteur de recherche et interface utilisateur, tout en répondant pleinement aux objectifs pédagogiques du projet.