Preprocessing data

* ~~Importer données~~ [~~Mathias Barbosa~~](mailto:barbosamat@cy-tech.fr)
* ~~Split les colonnes tag~~ [~~Justine Ribas~~](mailto:ribasjusti@cy-tech.fr)
* ~~Data exploration~~ [~~Justine Ribas~~](mailto:ribasjusti@cy-tech.fr)
* Preprocessing pour les synopsis (tokenisation, stopwords, …) [Mael Thire](mailto:thiremael@cy-tech.fr)
* Calcul de la distance (distance synopsis + tags) [David Lasgleizes](mailto:lasgleizes@cy-tech.fr)
* Faire le scrapping des années des films [David Lasgleizes](mailto:lasgleizes@cy-tech.fr)

Workflow :

* ~~Interaction chatbot (si y a un mot clé je te fais un prompt sinon dit ce que tu veux)~~ [~~Mathias Barbosa~~](mailto:barbosamat@cy-tech.fr)
* Prendre le film et calculer les distances -> donner les 10 meilleurs

Ressources :

* [Database](https://www.kaggle.com/datasets/cryptexcode/mpst-movie-plot-synopses-with-tags)
* [Kaggle - Movie Recomendation Netflix](https://www.kaggle.com/code/radiantmohit/movie-recomendation-netflix/notebook)
* <https://www.kaggle.com/code/rounakbanik/movie-recommender-systems>

Deadline :

1er check : week end du 25/26 Novembre

Rendu : 10/12/2023

#Fonction principale qui récupère les documents et les met dans le dossier Data

#Installation des packages

import requests

import html5lib

from pdf2jpg import pdf2jpg

import bs4

import sys

import urllib

import os, shutil

from bs4 import BeautifulSoup

#print(soup.prettify())

#print(soup.find\_all("a", href = True))

#print(r.content)

#Mise en place de la page et des pages souhaitées

URL = "https://factpages.npd.no/en/wellbore/PageView/Exploration/All"

URL\_seeked = "https://factpages.npd.no/en/wellbore/PageView/Exploration/All/"

searched\_link = [] #tableau des liens de tous les puits

completion\_link = [] #tableau des liens des docs pdf

def scraping() :

#Récupération de la page factpages

r = requests.get(URL)

soup = BeautifulSoup(r.content, 'html5lib')

#Boucle de recherche des pages souhaitées

for link in soup.find\_all('a', href = True): #pour tous les liens de la page

if (URL\_seeked in link.get("href")): #on prend seulement les liens souhaités

searched\_link.append(link.get("href"))

print(len(searched\_link))

print(searched\_link)

#for i in range(len(searched\_link) -1):

for i in range(10) :

print("i", i)

URLi = searched\_link[i] #on prend tous les liens

p = requests.get(URLi)

soupi = BeautifulSoup(p.content, 'html5lib')

#cpt = 1

for linki in soupi.find\_all('a', href = True):

if("COMPLETION\_LOG" in linki.get("href")): #que les liens avec COMPLETION\_LOG

completion\_link.append(linki.get("href"))

pdf= open("/content/drive/Shareddrives/DataBattle/Data/Docpdf/CompletionLog"+str(i)+".pdf", 'wb') #télécharger doc pdf à partir d'un lien

response = requests.get(linki.get("href"))

pdf.write(response.content)

#cpt = cpt+1

pdf.close()

inputpath = r"/content/drive/Shareddrives/DataBattle/Data/Docpdf/CompletionLog"+str(i)+".pdf"

outputpath = r"/content/drive/Shareddrives/DataBattle/Data/Essaipdfimage"

result = pdf2jpg.convert\_pdf2jpg(inputpath, outputpath, pages="ALL")

#print(i)

print(completion\_link)

print(len(completion\_link))

return completion\_link

scraping()

applied sur un dataframe