Projet New York Times - Bootcamp Data Engineer Février 2023

Mickael Gaspar, Can Baskurt, Clément Guiraud

Github : https://github.com/dst-nynews/dst-nynews
API New York TImes : https://developer.nytimes.com/apis
Doc de suivi projet: Suivi de projet
Documentation des fonctions: Fonctions du projet

Rappel sur les objectifs du projet et les données utilisées

Le projet que nous souhaitons mettre en place se fixe 3 objectifs métiers différents pouvant être, ou non, réalisés en fonction du temps et de notre avancement.

Nous souhaitons tout d'abord proposer un *dashboard* permettant de trier, filtrer et rechercher les éléments essentiels des articles et de la sémantique proposés par le New York Times. Il s'agira de permettre à l'utilisateur d'obtenir facilement des informations sur ce que propose le journal.

Dans un deuxième temps, il nous semblerait intéressant de mettre en relation les données sur le COVID proposées par le New York Times, et si le temps nous le permet par d'autres sources, avec les articles publiés par le New York Times. Cette comparaison pourrait permettre, notamment, d'avoir une meilleure compréhension de comment l'évolution réelle de la pandémie (évaluée par les données covid brutes) et sa retransmission dans un média grand public ont pu, ou pas, évoluer dans le temps.

Enfin, si le temps nous le permet, nous souhaiterions mettre en place une application web sous forme de mini-jeu de prédiction d'articles à succès. Il s'agira de permettre aux utilisateurs de prédire, parmi un set d'articles publiés par le New York Times, lequel sera présent dans les données "most popular" que propose le journal. La proposition de l'utilisateur pourra être comparée avec celle d'un algorithme de *Machine Learning* pour savoir si l'utilisateur est "meilleur ou moins bon qu'une intelligence artificielle".

Rappel sur les livrables 1 et 2

A la fin de notre premier livrable nous présentions les suites du projet comme suit :

La prochaine grande étape à mettre en place est le choix du type de stockage qu'il nous faudra utiliser. Ce choix est à la fois technique (BDD SQL ? NoSQL ?) et pratique (stockage en local par chaque membre du groupe ? stockage sur un VM? stockage dans le cloud ?).

La solution vers laquelle nous nous orientons est celle d'un stockage en local avec l'utilisation de scripts python assurant la similarité des données exploitées chez chaque membre du groupe.

Comme nous le notions dans le 2eme livrable, le choix des outils s'est finalement porté sur une base de données NoSQL, MongoDB dans sa version cloud Atlas, pour les données directement issues de l'API du New-York Times, et une base de données SQL, PostgreSQL, pour les données du Covid. Le peuplement de ces bases de données se fait à l'aide de scripts pythons pour l'ensemble des API du New-York Times que nous avons sélectionné.

Le livrable n°2 ouvrait des perspectives sur les différents cas d'usage que nous souhaitions mettre en place, autrement dit sur les différentes manières de consommer les données que nous imaginions.

Ce troisième livrable présentera les choix effectués quant à la consommation des données issues des 3 API du New-York TImes que sont Article Search, Semantic et Most Popular ainsi que des données sur le Covid fournies par le journal.

Cas d'usage N°1 : Afficher les articles les plus populaires

La première utilisation des données que nous avons récoltées provient de l'API Most Popular. En effet, cette API propose à celui qui la requête des informations sur les articles considérés par le journal comme étant les plus populaires où la notion de popularité est définie soit par le nombre de partage par mail, soit sur facebook ou enfin par la visibilité de l'article sur le site du New York Times.

Notre premier cas d'usage consiste donc à permettre à l'utilisateur d'afficher, à l'aide d'un calendrier et d'une liste déroulante, les articles les plus populaires pour une date donnée. Une fois la liste affichée, l'utilisateur pourra consulter l'article désiré sur le site du journal puisque les résultats sont liés à une URL renvoyant sur la page de l'article.

Ce *use case* s'appuie sur le framework streamlit.

Cas d'usage N°2 : affichage des informations sur le covid

Le deuxième cas d'usage que nous avons mis en place concerne les données covid proposées par le New-york Times. L'objectif final de ce cas métier est de présenter un graph permettant de comparer le nombre de cas et le nombre de morts du Covid avec le nombre de publications d'articles liés au Covid. L'idée derrière ce cas d'usage était de pouvoir comparer l'importance donnée au Covid par un journal comme le New-York Times avec le nombre réel de cas, et de décès, effectivement associés à la maladie. Cela permettrait de répondre à des questions telles que : l'importance donnée à la maladie a-t-elle baissé avec le temps ? Le nombre d'articles publiés est-il corrélé avec l'avancée réelle de la maladie ?

Par simplicité, nous avons séparé ce cas d'usage en 2. Il s'agit tout d'abord d'afficher un tableau présentant, en fonction d'une date de départ et d'une date de fin rentrées par l'utilisateur, d'afficher par states et au total le nombre de personnes atteintes par la maladie et le nombre de décès sur cette plage de temps.
Une fois cette étape réussie, et si le temps nous le permet, nous souhaitons afficher sur un graph 3 courbes en fonction du temps : la 1ere représentant le nombre de cas de Covid, la 2ème représentant le nombre de décès et enfin la 3ème représentant le nombre d'articles

liés à au mot clé “Covid” retourné par l’API Article Search et peuplant actuellement la base de données NoSQL.

Si le temps le permet, nous avons aussi pensé aux éléments suivants:

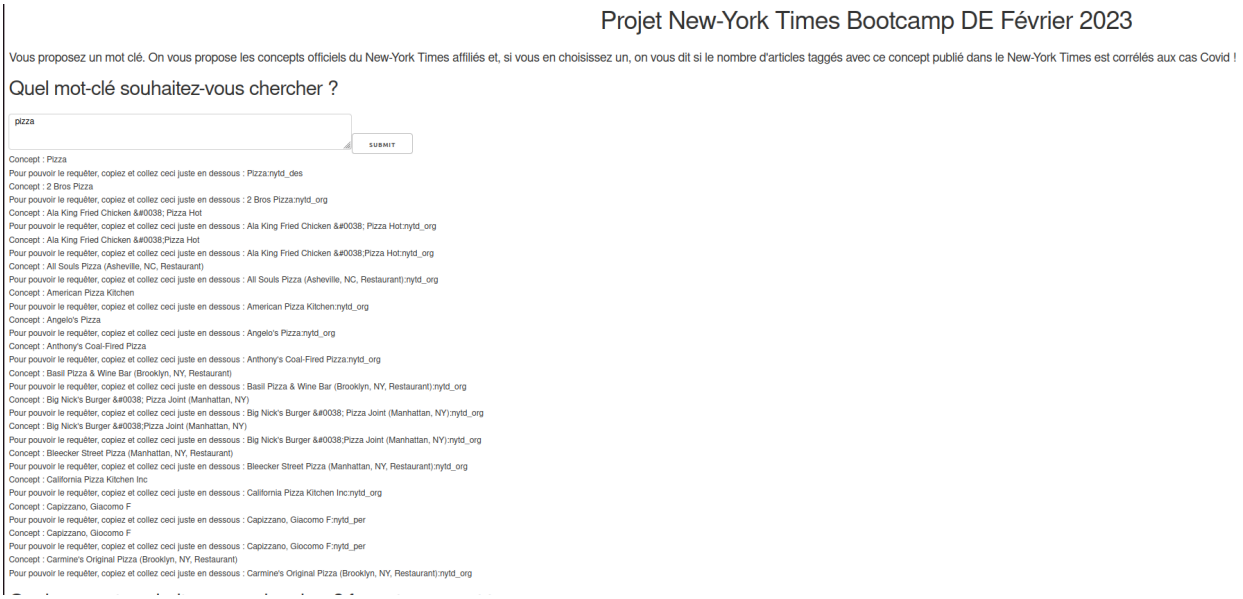
- Mask use: moyenne par states des stats de mask use: never, rarely, sometimes, frequently, always
- Obtenir à partir de 1 county (= 1 county name et 1 state name en input): le nbr total de cases, deaths et pour ce même county, afficher les stats de mask use pour voir l'incidence de l’utilisation des masks sur le nbr de morts et de cases covid.

Ce cas d’usage est réalisé à l’aide du *framework* python Dash et si nous avons des difficultés quant à la réalisation de ce cas d’usage ce n’est pas tant par son aspect *data engineering* que par l’utilisation de ce *framework*.

Cas d’usage N°3 : De la chaîne de caractères aux informations sur un concept officiel du NYT

Le dernier cas d’usage que nous avons réalisé s’appuie sur les données issues de l’API Semantic proposée par le New-York Time. Il s’agit de permettre à l’utilisateur de proposer une chaîne de caractère de son souhait et d’obtenir les informations liées au concept officiel se rapprochant, selon lui, le plus de cette chaîne de caractère.
Là encore, ce cas d’usage est découpé en 2 étapes distinctes. Tout d’abord, il s’agit de proposer à l’utilisateur une liste de concepts officiels, et leur type affiliés, lié à la chaîne de caractère qu’il vient de rentrer. Une fois cette liste de concepts correspondant à la chaîne de caractère qu’il souhaite, l’utilisateur peut obtenir des informations sur le concept de la liste qui lui semble correspondre au mieux à sa chaine de caractère.
Lui seront ainsi accessibles les informations suivantes :

1. Le nombre d’articles taggés avec ce concept par le New-York Times
2. La date de création du concept
3. Le statut du concept (actif ou inactif)
4. Une liste d’articles et leur URL liés au concept



Première étape du cas d'usage n°3

2ème étape du cas d'usage n°3

Perspectives

Les cas d'usages des données extraites des APIs du New-York Times maintenant réalisés il nous reste maintenant à les mettre en production. Pour cela, nous nous orientons vers la ecréation d'un API à l'aide de FastAPI. L'idée est d'avoir une liste de endpoints par tables (ou collection) des bases de données nous permettant de mettre en œuvre les cas d'usages ici présentés. Une fois cela réalisé, il nous faudra containeriser notre travail. Si le container de la base de données SQL est d'ores et déjà fonctionnel, nous souhaitons proposer deux autres containers : le premier contenant l'API et le second les différents *frontends*.