Rapport de Stage de fin de tronc commun

DAMOUR Jeremy

Entreprise: Factonics

16 septembre 2019 - 24 janvier 2020

FACTONICS

sujet:

Développement d'un module de déploiement automatique de modele de Machine Learning et versionning.

Le sujet du stage consiste au développement et déploiement d'algorithmes de machine Learning, sur la plate-forme de l'entreprise. Cette plate-forme est destinée à être consulté par le different client, pour qu'ils puissent choisir l'algorithme qu'il désire.

Sommaire:

I. Introduction

Ce rapport présente les travaux réalisés au cours du stage de fin de tronc commun à EPITA (École Pour l'Informatique et les Techniques Avancées), réalisé de début septembre 2019 à fin janvier 2020. J'ai souhaité trouver un stage « long » adapté à mon niveau en sortie d'ING1 et correspondant à mon projet professionnel. Ce stage est mon premier stage en informatique et est le stage le plus long que j'ai fait. Il est vraiment bien d'avoir des stages longs pour pouvoir bien se familiariser avec le sujet et avoir de réelles responsabilités dans l'entreprise. Ce stage permet de mettre en application les méthodes pour coder apprise dans l'école.

Ce stage devait me permettre de découvrir la data-science et voir si cela me plaisait, pour m'aider dans mon choix de majeur. Ce domaine me plaît réellement et cela m'aide à voir que je veux devenir un ingénieur dans le domaine de la data-science.

Il conforte donc mon choix de majeure étant SCIA (data-science et intelligence artificielle) ou IMAGE. Je suis donc heureux aujourd'hui d'aller dans la majeure IMAGE. Le domaine de la data-science est d'autant plus intéressant, car il est constamment en évolution.

Mon stage s'est déroulé chez Factonics une start-up se positionnant sur le marché de la Machine Learning en offrant à des clients très orientés métiers des algorithmes prêt à l'utilisation. Factonics a pour but de proposer dans quelques mois une plate-forme ouverte à tous les secteurs d'activité.

Mon maître de stage n'est autre que le fondateur de l'entreprise Charles Dadi. Le but de ce stage était de pouvoir comprendre le fonctionnement et réaliser un algorithme de Machine Learning et pouvoir le valider. J'ai aussi pu apprendre comment déployer différentes applications et algorithmes de Machine Learning.

Le stage était mon premier stage de plus d'un mois, il m'a notamment fait découvrir plus en détail le monde de l'entreprise, en particulier ici celle des start-ups (Factonics étant une start-up). J'avais une appréhension quant aux start-ups, je pensais que ce genre d'entreprise n'était pas fait pour moi en vue de l'organisation peu existante, du boulot avec les dates de rendu étant très courtes. J'ai été heureux de faire mon stage dans une start-up comme Factonics, car je n'avais pas le temps de m'ennuyer, il y avait toujours du

travail. Les objectifs ainsi que les dates de rendu étaient assez précis. Et j'ai vraiment pu voir l'utilité du projet sur lequel j'ai travaillé.

Factonics étant une start-up avec ses propres locaux, il a été simple de tisser des liens avec d'autre personne de l'entreprise et ainsi avoir une bonne ambiance de travail. Pendant mon stage, il y avait un Brieg Oudea un stagiaire venant d'EPITA qui faisait tout comme moi son stage de tronc commun.

Je pense donc qu'être dans une start-up est idéale pour gagner de l'expérience rapidement dans un domaine. Nous allons voir par la suite plus en détails, les objectifs de société, comment elle se compose.

Le sujet de mon stage était : « Développement d'un module de déploiement automatique de modèle de Machine Learning et versionning. ». J'avais donc comme objectif de pouvoir crée un algorithme de machine learning et le déployer. J'ai commencé par découvrir, puis la réaliser un algorithme de machine learning. Et à la fin de mon stage, le versionning et déploiement d'un algorithme.

Mon maître de stage a réussi à bien segmenter, le travail à me donner. On faisait un point sur le travail que j'avais fait trois fois par semaine. Ces points ont été assez espacés pour me laisser le temps pour me documenter et travailler sur les différentes tâches. Dans ces points, on faisait la validation ensemble sur le travail que j'avais effectué et discuté des éventuelles améliorations que je pourrais intégrer (s'il y avait du temps restant). J'ai eu une grande liberté sur l'implémentation pour arriver aux objectifs fixés.

J'ai aussi pu assister à différentes réunions téléphoniques avec le client, mais aussi allez directement dans leurs locaux. Et donc voire à qui mon travail allait servir et voir leur attente.

J'ai souvent travaillé en groupe avec un stagiaire, quand les délais étaient courts. Sur la fin de mon stage, j'ai contribué au déploiement de son algorithme de Machine Learning. J'ai aussi dans ce stage appris à travailler en télétravail.

Le sujet principal a été la réalisation d'un algorithme de Machine Learning pour un client se positionnant dans le domaine de la parfumerie. Ce projet consiste à récupérer des retours des utilisateurs sur des parfums et à en extraire les différentes caractéristiques du parfum vu par les utilisateurs du produit. Et donc extraire avec ses caractéristiques les points forts et faibles du parfum en question.

Cet outil est utilisé par les clients pour pouvoir créer de meilleurs parfums, avec les retours des utilisateurs. Le projet était déjà existant dans l'entreprise, je n'ai donc pas commencé de rien.

Néanmoins, il a fallu optimiser le code et document certaines fonctions. Ainsi qu'implémenter de nouvelle fonctionnalité au projet.

Pour ce projet, j'ai donc dû travailler sur :

- la récupération de données de différentes sources (scraping)
- le matching entre les différents produits de toutes les sources
- améliorer les performances du modèles servant avec trouver le sentiment d'un commentaire
- construire un algorithme qui construit des groupes de mots liés entre eux (pour trouver les différentes catégories dans les commentaires)
- regrouper toutes ces informations pour obtenir les opinions issues des commentaires.

Vous allez par la suite pouvoir voir plus en détail l'entreprise. Le fonctionnement du projet principal auquel j'ai été affecté. Et mes taches secondaires dans l'entreprise.

II. Présentation De L'Entreprise

a. Le secteur d'activité

FACTONICS est une start-up qui a pour objectif de démocratiser l'accès au Machine Learning.

L'entreprise est tournée vers l'économie de la data et vise à améliorer l'exploitation des données de leurs clients et leur propose de nouvelles sources de valeur en exploitant de manière innovante les techniques d'analyse.

Elle souhaite apporter aux équipes métiers des algorithmes de Machine Learning à utiliser au quotidien.

Factonics a su se faire connaître en s'adressant à des profils non-techniques pour qui l'IA n'était pas accessible sans un lourd coût de mise en place.

b. L'entreprise

La start-up est incubée à CentralSupelec. Elle a été fondée en 2016 par Charles Dadi. Les locaux de la start-up se trouvent dans le 19 ème arrondissement de Paris.

Factonics est aujourd'hui composé à l'heure actuelle de 7 salariés et 4 stagiaires. On y trouve beaucoup de profil orienté dans la Data Science, mais aussi Juriste, Account Manager, développeur et designer…

L'entreprise est divisée en 3 pôles :

- Pôle R&D en Data Science autour des algorithmes alimentent la plate-forme (principale activité)
- Pôle développement, qui se charge de la maintenance, de l'ajout de nouvelle fonctionnalité et l'intégration des nouveaux algorithmes a la plateforme. Ce pôle est étroite relation avec le précédent.
- Pôle management et relation client qui s'occupe de la vie interne et externe de l'entreprise (en assurant mise en place de conférence, newsletter, article sur Médium, appelle avec les clients, proposition commerciale…)

Voici les principales personnes composant l'équipe de Factonics:

Eugénie - Account Manager

Charles Dadi - Machine Learner

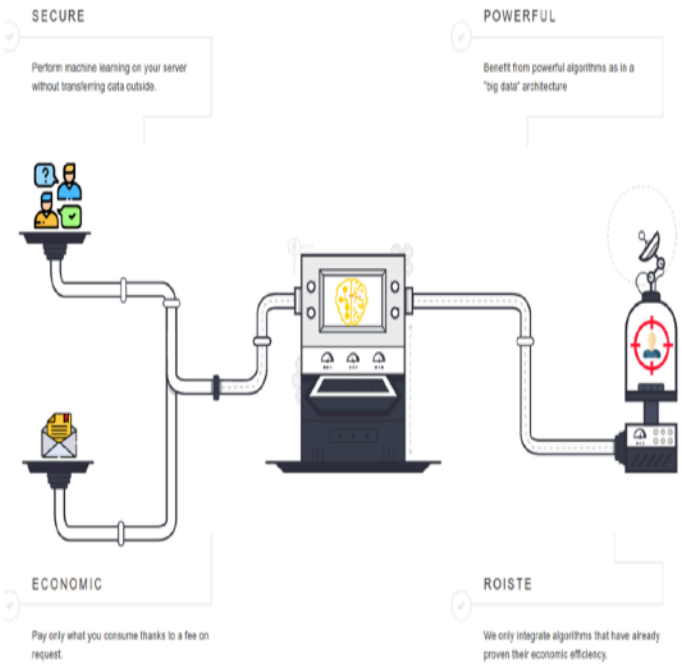
Rebecca - Juriste Data Protection

Huy Dan - Data Engineer

Xinze - Data Scientist

c. Le service

Factonics a créé initialement, un outil se nomme « Top-model » qui est un catalogue d'algorithmes de Machine Learning prêt à l'emploi proposer à des clients dans le secteur de la mode.



Explication de top-model

Sur cette base, l'entreprise a décidé de développer une plate-forme sur le même concept, mais cette fois ouvert à tous les secteurs d'activité. Cet outil a pour but principal de pouvoir ouvrir et lancé l'IA pour tous.

À travers cette plate-forme l'objectif est d'apporter des algorithmes utiles aux équipes métiers pour permettre l'utilisation la plus simple possible de l'IA quel que soit le domaine (banque, beauté, santé, marketing…) Factonics veut que cette application soit simple d'utilisation, pour que les clients peu familiers avec le domaine puissent l'utiliser facilement. Mais aussi transparente pour les clients soucieux de l'utilisation de leur donnée.

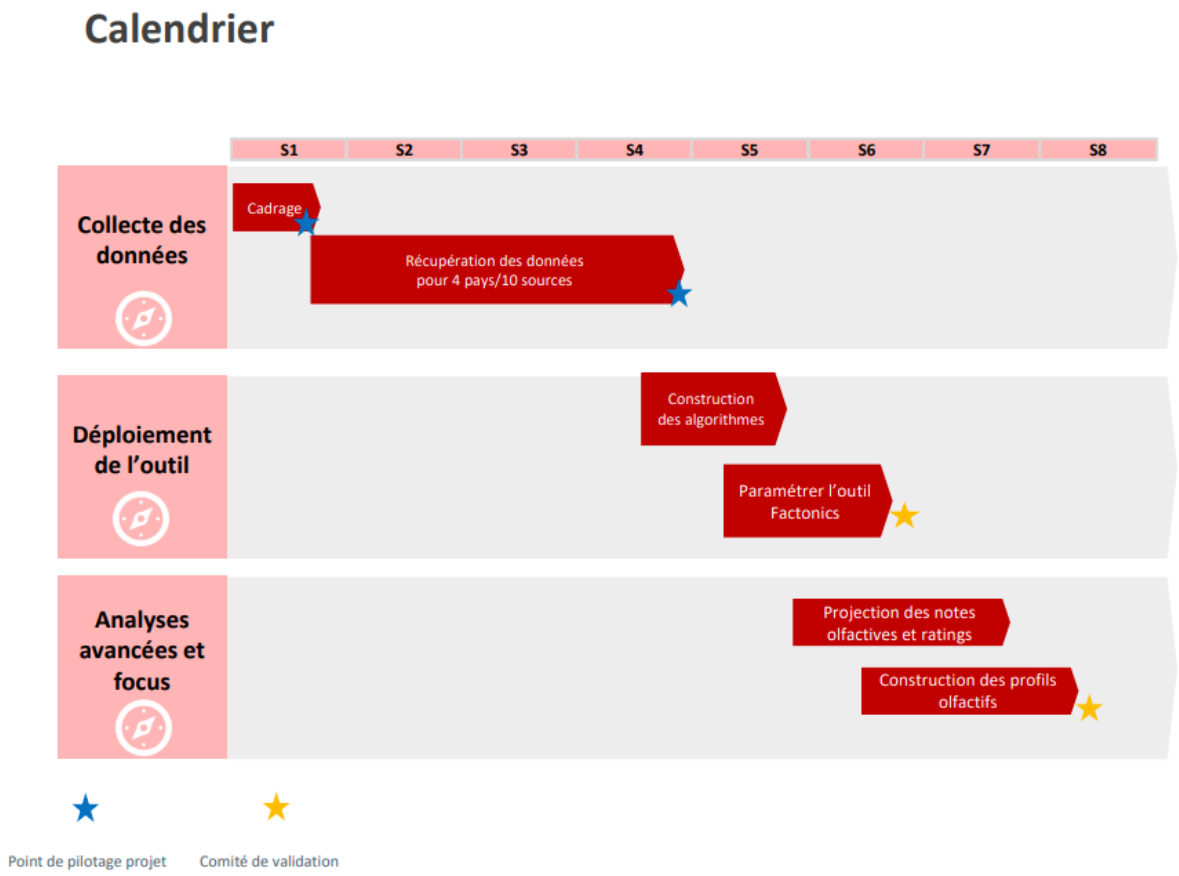
Cette application permet donc aux clients de pouvoir exploiter l'IA et ainsi avoir de nouvelle recommandation pertinente pour les permettre d'optimiser et s'améliorer dans leur domaine d'activité. La solution sera disponible sur le site de Factonics avec plusieurs formules.

Factonics ne semble pas aujourd'hui avoir de concurrents directs, car les solutions sur le marché sont destinées à des profils plus techniques en proposant des outils destinés à des data Scientists confirmé. Là où Factonics au contraire s'adresse à des équipes techniques n'allant pas beaucoup d'expérience dans le domaine. Les conçurent, qui ne sont pas directs à Factonics sont des entreprises tels que : Dataiku, Algorithmia, prévision IO…

d. Le positionnement du stage dans les travaux de l'entreprise

J'ai pour but dans l'entreprise de faire avancer cette plateforme en développant un ou plusieurs algorithmes de Machine Learning.

Mon stage a eu pour but essentiellement de continuer un projet de data science, déjà présent dans l'entreprise, pour un client se positionnant dans le marché de la parfumerie.



Calendrier du projet

Avec ce projet, j'ai pu acquérir de l'expérience dans l'utilisation des différentes librairies. Ce projet a demandé une grande responsabilité, car il était important pour le développement de l'entreprise. Dans ce projet, il a fallu collecter, l'analyser, traiter et la valider les données.

Ce projet ne reste pas spécifique à un client, il peut être réutilisé.

J'ai aussi travaillé en binôme (avec un stagiaire d'EPITA), sur un autre projet sur le dernier mois de mon stage. J'ai travaillé à mettre en production un autre projet de l'entreprise.

J'ai aussi appris beaucoup sur le déploiement d'application avec Docker, docker-compose.

III. Travail Effectué

a. Le cahier des charges (C.D.C.)

1) But Général

Le but est de pouvoir crée un algorithme de Machine Learning, pouvant servir à des clients de l'entreprise. Ceci passe par la compréhension des différentes étapes de la conception d'un tel algorithme.

Le projet où j'ai été affecté avait pour but de me former aux Machines Learning en me donnant un projet en partis déjà fait. Ce projet comportait une très grande partie de récolte des données par du scraping sur plusieurs sites.

Dans l'entreprise j'ai beaucoup travaillé sur un projet en particulier, mais j'ai notamment pu aider sur d'autres projets quand des dates de rendu étaient serrées. J'ai notamment fait du déploiement de plusieurs applications avec docker.

J'ai eu comme objectif de pouvoir terminer le projet que l'on m'a donné. En respectant bien chaque rendu. Et de pouvoir aider au déploiement d'application. Ou déploiement d'autres projets en production.

2) Explication détaillée des résultats à obtenir

Pour le projet principal, j'ai dû passer par toutes les étapes du Machine Learning:

- récupération de données (scraping de produit et de retour utilisateur sur plusieurs sites de beauté)
- réunir les données des produits de différentes sources avec le moins de perte et la meilleure correspondance tout en retirant les produits qu'on ne voudrait pas (ex: si on cherche des parfums, mais qu'on a des fonds de teint dans nos données).
- savoir faire des groupes de mots pour pouvoir établir des catégories.
- savoir faire un prétraitement des données et faire une validation ces nouvelles données.
- savoir entraîner plusieurs modèles pour pouvoir déterminer les sentiments des commentaires (et savoir valider les modèles).

- pouvoir construire une méthode ensembliste et valider les performances de cet ensemble.
- regrouper toutes les données pour avoir l'opinion d'un commentaire, et donc enrichir les données de base.

J'ai aussi eu des taches plus secondaires telles que :

- pouvoir déployer des applications (comme mongoDB, elastics search, kibana, Logstash, mongstash, airflow…) avec docker-compose
- aider au déploiement d'un autre sujet dans l'entreprise avec Airflow
- aider sur les autres projets qui ont une date rendue très proche

b. Compte-rendu d'activité

1. Axes d'étude et de recherche choisis

Mes différents axes d'études ont été :

- la récupération des données par scraping
- la gestion d'une base de donné mongoDB
- faire correspondre différents objets (fuzzy matching)
- le NLP (Natural Language Processing)
- savoir faire des groupes de mots (clustering)
- la création et la validation d'un modèle
- le déploiement d'application par Docker
- utiliser un workflow comme airflow pour mettre en production un algorithme de machine learning

2. Déroulement concret des études, expérimentations

i. Autoformation

J'ai commencé par voir les différentes étapes de conception et déploiement d'un algorithme de machine Learning, ainsi que des outils de validation.

Je me suis donc documenter sur:

- le scraping de donnée
  - Requests ("Cf. annexes page - 46").
  - avec la librairie Beautifulsoup ("Cf. Annexe page - 46").
  - le framework scrappy ("Cf. Annexe page - 46").
- mongoDB (car toutes les données sont stockées dans cette base de données) et pymongo (car les scripts dans l'entreprise sont faits en python) ("Cf. Annexes page - 47").
- le NLP (Natural Language Processing)
  - POS-Tagging / NER / NLG (non utilisé)
- processing de donnée:
  - Stemming (non utilisé)
  - Lemmatisation [8] (avec SpaCy ("Cf. annexes page - 47").)
- lecture sur l'utilisation de différent model:
  - fastText ("Cf. annexes page - 47").
  - différente régression (ex: régression logistique)
- TSNE ("Cf. annexes page - 48").
- word2vec ("Cf. annexes page - 48").
- validation des modèles
  - matrice de confusion
  - courbe de ROC
- utiliser Jupyter notebook ("Cf. annexes page - 45").
- utiliser screen ("Cf. annexes page - 45"). pour pouvoir faire tourner des scripts sur une machine même quand la connexion ssh est fermé
- découverte de docker ("Cf. annexes page - 49") et docker compose
- airflow ("Cf. annexes page - 49")(une plateforme de management workflow)
- Boto3 ("Cf. annexes page - 49")(pour pouvoir gérer un bucket S3 dans mon cas)

J'ai eu au début de mon stage un projet utilisant git ou un projet avait déjà été commencer, je devais le continuer.

Même si je travaillais la plupart du temps sur ce repository git, il a été important de travailler sur des branches différentes selon la partie du projet. Ses branches étaient ensuite « merge » à la branche maître (branch master) en utilisant les pulls requests.

Ce projet se nomme « trackmarket », ce projet a été créé pour les clients souhaitant récupérer et enrichir les données de plusieurs sites publics où on va récupérer les produits et les retours des utilisateurs. Ils ont donc un retour des clients sur leurs produits.

J'ai donc dû mettre à jour ce projet pour un client se positionnant sur le marché de la parfumerie. De base, le projet devait être assez complet, je pensais n'avoir qu'à faire les récupérations des données sur différents sites de parfum. Au final, j'ai changé toutes les grandes parties du projet pour avoir de meilleures performances.

Après une semaine du lecture de code de ce projet pour bien le comprendre, ainsi que la lecture de documentation.

Avec les premiers soucis d'environnement python que j'ai eu avec un « requirements.txt » incomplet, j'ai décidé d'utiliser dans ce projet un pipenv ("Cf. annexes page - 45").

Lors de ce projet beaucoup de point on était fait avec mon maitre de stage.

Ces points pour but de :

- valider mon travail
- faire des revues du code que j'avais pu écrite
- aider à la réflexion des problèmes rencontrés
- valider les différentes données
- faire du pair programming (programmation à deux)
- fixer les nouveaux objectifs

Ces points on été très important tout au long du stage. Ils étaient assez réguliez dans le temps et assez espacer, pour pouvoir avoir un travail efficace.

1. Activité principal

i. Collecte des données

Il a fallu commencer par la collecte des données avec du scraping sur différents sites de parfum, tel que:

- Fragrantica (www.fragrantica.fr)
- Sephora (www.sephora.fr/shop/parfum-c301/)
- Beauté-test (www.beaute-test.com/parfums.php)
- Marionnaud (www.marionnaud.fr/parfum/c/P0000)
- Amazon (www.amazon.fr)

Certains scrapers étaient déjà écrits, comme celui de beauté-test, Sephora et Fragrantica. Le seul qui était encore viable pour être utilisé était celui de Fragrantica. Tous les autres, on dut être réécrit.

scraping

J'ai donc essayé d'écrire le scraper le plus générique possible. Le scraper devait être assez générique pour :

- les personnes devant écrire par la suite d'autres scrapers
- pour moi, car je devais réécrire plusieurs scrapers

La récupération des données sur des sites internet se fait en 2 étapes :

- le crawling[4] (ou on va indexer les liens où il y a des données à récupérer.)
- le scraping[4] (ou on va récupérer les données qu'on veut pour les stocker.)

Pour faire le scraper le plus générique, j'ai utilisé la programmation orientée objet, en utilisant l'héritage entre les classes, mais aussi les générateurs python pour pouvoir changer les pages HTML au fur et à mesure.

J'ai donc dû implémenter les deux classes :

- Produit à analysé (Product_parsing)
- Retour de l'utilisateur à analysé (Review_parsing)

Ces deux classes prennent à l'initialisation la page HTML a analysé. On a donc dans ces 2 classes des méthodes qui doivent absolument être implémentées, pour analyser de l'HTML. Elles ont une méthode pour pouvoir générer la structure à envoyé dans la base donnée mongoDB.

MongoDB est une base de données No-SQL. Il n'y a pas de validation de données avant l'insertion dans une « collection » du côté mongo.

Il faut donc du coté python faire cette validation sur :

- les noms des clefs
- le type des valeurs (int, string, bool, datetime…)

On a donc pour cela un wrapper[17] sur les fonctions de pymongo, ou on ajoute un schéma de validation des données (on utilise voluptuous ("Cf. Annexes page - 47")), lors de l'insertion et l'update des données dans la base.

```
productSch = Schema({
    Required('idProduct'): Any(str),
    Required('name'): Any(str),
    Required('brand'): Any(str),
    Required('retailer'): Any(str),
    Required('catName'): Any(str),
    Required('currentPrice'): Any(float, int, None),
    Required('createdAt'): Any(datetime.datetime, datetime.date, None),
    Required('updatedAt'): Any(datetime.datetime, datetime.date, None),
    Required("urlProduct"): Any(str),
    Required("urlImage"): Any(str, None),
    Required("urlRetailer"): Any(str, None),
    Optional('otherInfos'): Any(dict),
    Optional('specificInfos'): Any(dict)
})
```

schéma pour les produits

Par la suite, j'ai eu des problèmes, avec certains sites qui mettent en place des techniques d'anti scraping.

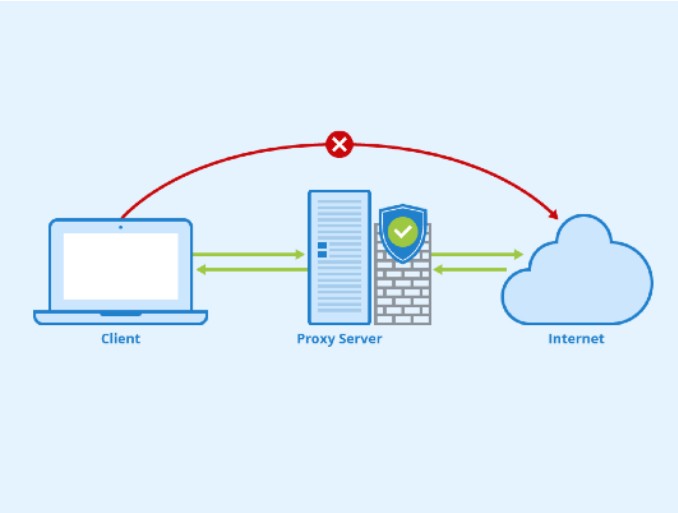
Il avait comme technique d'anti-scraping :

- bien vérifier les en-têtes surtout le header User Agent
- bloquer les IP pour une certaine duré
- faire attendre l'utilisateur pendant un certain temps
- page blanche

Pour réagir à cela, j'ai donc dû mettre en place plusieurs techniques pour pouvoir continuer à faire un scraping efficace.

Les solutions ont été :

- mettre à jour le header User Agent régulièrement
- passer par TOR [13]
- passer par des proxy [14]
- marquer les pages n'ayant pas réussi le scraping pour recommencer plus tard
- utiliser scrapy (avec les méthodes citées précédemment utiliser pour Amazon)



Fonctionnement d'un proxy

Ces 3 premières techniques, ont très bien fonctionné sur les différents sites qui nous bloquaient l'IP. Par contre pour Amazon à un moment malgré les proxy, il arrive tout de même à retracer l'IP de base. C'est pour cela qu'on a utilisé le Framework de scraping : Scrapy qui a pour avantage de faire les requêtes avec des thread et de recommencer une requête si une erreur se produit. On a donc implémenté les proxy, et recommencer le scraper d'Amazon sur le framework.

Il y a différentes stratégies pour effectuer le scraping dans notre cas.

La première stratégie consiste à récupérer d'abord le Product, puis directement les Reviews associer.

La seconde stratégie consiste à aspirer tous les Products, puis aller chercher en base les produits pour aller prendre les Reviews. Et marquer par la suite le Produit lorsque le scraping des Reviews est terminé.

Il y a une dernière aspirée qui consiste à récupérer les informations produites dès l'étape de crawling (si on a la plupart des informations dont a besoin). Puis comme la méthode précédente récupérer les reviews par la suite.

Pour tous les sites que j'ai récupéré les données, j'ai utilisé la première méthode.

Pour Amazon lors de l'écriture du nouveau scraper un utilise la troisième méthode, mais il va nous manquer avec cette méthode les marques de chaque produit. On va donc pour cela faire un « fuzzy matching » (technique expliqué par la suite) sur les autres produits qu'on a déjà récupérés. Pour déterminer la marque à mettre sur le produit en base. Amazon a aussi une particularité de boucler sur les pages, il faut donc indiquer la page maximale. Il faut aussi isoler les pages que l'on veux avec ceux qui appellent des « nodes », qui est un identifiant de catégorie.

Le site Fragrantica a été aspiré par un script déjà fait qui était très bien écrit. Ce site ne contient pas de Review, mais juste des products. Par contre le scraping de ce site a pris près de 3 semaines, car à un moment, si on fait trop de requête le site nous dit d'attendre avant de faire une autre requête. Il y a un temps d'attente de plus ou moins 5 min et cela, tous les 10 requêtes à peu près. On a donc avec Charles mon maître de stage a décidé de faire le scraping sur plusieurs machines pour avoir plusieurs IP. Il m'a donc fallu découper le scraping en plusieurs groupes sur plusieurs machines différentes.

J'ai pour cela utilisé « argparse ("Cf. Annexe page - 46") » , pour ajouter des flags comme :

- le nombre machine total
- le numéro de la machine utilisé
- refaire les tests de toutes les pages depuis le début

Il m'a donc fallu déployer le script sur plusieurs machines et de monitoire (dans un «GNU screen ») le scraping, pour voir son avancement, et l'éventuel crash des scripts.

A la fin de chaque scraping, j'ai fait avec mon maître de stage une validation, pour cela on vérifie :

- qu'il n'y a pas de doublon en base
- que les volumes sont bons

En parallèle du scraping j'ai testé le code qui m'était donné avec les données que j'avais commencées à récupérer et faisais une re-factorisation de certaines parties. Et aussi pouvoir identifier toute la pipeline du projet.

iii. Fuzzy Matching

Après la collecte, il a fallu uniformiser la donnée pour cela, on a utilisé une technique étant le « fuzzy-matching » (correspondance floue en français).

Notre fuzzy matching ce fait entre les produits pour regrouper les produits similaires entre eux (que ce soit par site, mais aussi par le volume du parfum). Cela va nous permettre par la suite de grouper les Reviews d'un même produit.

Les différentes étapes vont être :

- choisir un site pivot
- appliquer un preprocessing aux données
- crée des règles de matching à partir des informations du produit en base
- faire des fichiers de match et de non-match pour faire une validation
- insérer en base les produits qui match

Ce script existait déjà, mais il avait des défauts d'optimisation, il prenait donc beaucoup de temps. J'en ai profité pour faire une refactorisation du code, ainsi que de l'optimisation. J'ai aussi ajouté plus de logs, et des bar de progression avec tqdm("Cf. Annexe page - 46").

Comme pivot a été choisi le site Fragrantica avec le client car :

- il répertorie la plupart des parfums (près de 55 000 parfums)
- il est maintenu par des professionnels de la parfumerie

On va commencer par ajouter à nos donné à traiter une liste de combinaisons étant, fait par différents préprocessing sur les marques et les noms des produits:

- on va découper la phrase en petit groupe de mots et appliquer des permutations entre eux. On limite le nombre d'éléments à 4 pour avoir que 24 permutations possibles (4!=24) Pour faire ces permutation nous utilisons une fonction de itertools ("Cf. annexes page - 48").
- retirer les caractères spéciaux en utilisant « \W », contenue dans la Library regex de python re("Cf. annexes page - 45")
- retirer les digits
- retirer les accents
- retirer les mots inférieurs à 2 lettres
- retirer des mots définis (comme parfums, eau de Cologne …)
- retirer les mots avec 2 ou 3 digits suivis par des mots de 2 lettres (ex: 20 ml) avec une regex
- pour les marques on ajoute les abréviations (ex: Yves Saint Laurent devient YSL)
- pour les noms, on va retirer la marque parfois contenue à l'intérieur (et bien faire attention quand la marque et le nom du produit sont identiques)

On va par la suite créer des règles de matching. On va commencer par des règles toutes simples ou on est sûr d'avoir un bon match, puis on va descendre en qualité. On va même ajouter des règles où en modifier pour pouvoir répondre à certains cas bien spécifiques, qu'on aura remarqué en analysant le fichier sorti de matching.

On a donc des règles qui utilisent sur les marques et les noms des produits :

- le matching complet
- des regex
- la liste des combinaisons décrite précédemment
- des distances (distance de leveinshtein, distance de Xinse qui est propre à l'entreprise…)

Lors de mon stage, sur cette partie « fuzzy Matching » j'ai pu créer de nouvelle règles mais aussi faire gagner en performances les règles déjà existantes. J'ai aussi ajouté un fichier statistique (que vous pouvez voir ci-dessous).

J'ai rendu le script de beaucoup plus lisible, en créant une class Matching ou est stocker toutes règles. Ainsi qu'une méthode permettant de choisir les méthodes que l'on voudrait appliquer.

J'ai aussi amélioré le temps du processus:

- en retirant les éléments déjà match, dans une règle n - 1 (ayant donc un matching plus fort)
- en arrêtant le matching si tous les éléments ont déjà été match
- en utilisant au maximum, les fonctions de la librairie pandas ("Cf. annexes page - 46"), pour avoir une bonne vectorisation.
- j'ai aussi ajouté des bars de progressions pour mieux voir l'état du script.
- ne pas recharger à chaque site à match le site pivot.

J'ai changé certains seuils (au niveau des distances), pour avoir des matchs assez précis et donc pas trop large.

Il était important de garder un assez gros volume, pour avoir assez de données pour satisfaire la demande du client. Mai aussi dans ce script d'avoir un match précis, car d'un point de vu métier, c'est primordiale. Si on n'est pas assez précis par la suite, nous allons avoir des produits mal identifié et donc pour le client cela va être une source d'erreur.

example de fuzzy matching

nom de Référence	marque de Référence	nom	marque	règle
cinema scenario ete	yves saint laurent	scenario ete cinema	yves saint laurent	rule_1
sauvage	christian dior	sauvage	dior	rule_3
adore	christian dior	adore	dior	rule_3
ivoire	pierre balmain	ivoire	balmain	rule_3

On peut voir ici à droite un bout d'un fichier statistique qui se trouve à la sortie d'un fuzzy matching récapitulent:

- les règles utilisées
- le nombre de produits qui ont été match
- le pourcentage de match sur la règle

beaute-test	rule_0	3158	53.65
	rule_1	277	4.71
	rule_2	226	3.84
	rule_3	263	4.47
	rule_3.5	583	9.9
	rule_4	123	2.09
	rule_5	88	1.5
	rule_6	34	0.58
	total	4752	80.73
marionnaud	rule_0	814	59.46
	rule_1	22	1.61
	rule_2	7	0.51
	rule_3	77	5.62
	rule_3.5	154	11.25
	rule_4	32	2.34
	rule_5	14	1.02
	rule_6	0	0
	total	1120	81.81
sephora	rule_0	1158	65.87
	rule_1	37	2.1
	rule_2	22	1.25
	rule_3	135	7.68

fichier statistique de sortie d'un fuzzy matching

ii. Preprocessing des Reviews

On doit par la suite commencer à traiter les reviews pour les utiliser dans certaines étapes plus loin.

Le préprocessing est une étape très importante quand on fait du NLP(natural language processing). Il permet d'avoir des données plus uniformes et donc plus facilement traitable en sortie.

Dans cette partie, on va chercher à passer d'une review à une séquence. Une séquence va être un bout de notre review. On va donc pour cela découper notre review en plusieurs séquences.

Ces séquences nous serviront par sa suite a construire une citation, qui contiendra:

- une fonctionnalité produit
- une opinion associé a la fonctionnalité
- un sentiment (positif ou négatif)

Il faudra bien faire attention à cette étape de préprocessing d'avoir un moyen de remonter à la phrase originale (surtout sur les mots qui auront changé de forme).

Voici donc le péprocessing que l'on va appliquer sur chaque séquence obtenue:

séparer en token et l'associer à un numéro

mettre tous les caractère en minuscule

retirer la ponctuation

retirer les accents

retirer les mots de moins de 2 lettres

retirer les mots qu'on ne veut pas (ex: des, le, la …)

correction orthographique

lemmatisation

On associe in token a un numéro pour pouvoir retrouver sa position initiale à la fin du préprocessing.

Il faudra pendant l'étape de la légalisation réunir tous les tokens et les passer dans le lemmatizer (SpaCy) et par la suite retrouver les tokens changés par le lemmatizer, et leur redonner leur numéro initial.

La correction orthographique que j'ai faite se base sur:

- un dictionnaire français
- tous les mots du corpus qui vont être pris à partir d'un certain seuil
- les notes olfactives récupérées dans Fragrantica

On va ensuite comparer les mots non trouvés dans ce dictionnaire pour avoir le mot le plus proche, en utilisant des permutations et des suppressions de lettres.

La lemmatisation est une technique de préprocessing très puissante, il va permettre de faire retourner un mot à sa racine, et il va supprimer des mots inutiles dans la phrase. Le lemmatizer que j'ai utilisé est SpaCy("Cf. annexes page - 47"), il est très long en processing, et parfois pas très fiable.

iii. Groupe de mots par thématique

Sur la recherche de Topic nous allons effectuer un Topic Modeling, j'ai travaillé en binôme avec l'autre stagiaire venant d'EPITA, car nous avons eu des délais assez serré. Sur cette partie nous devons faire un clustering sur les mots du préprocessing fait précédemment, pour obtenir par la suite les fonctionnalités du produit et les opinions. Et de créer un Dashboard pour que les clients puissent valider chaque cluster.

Cette partie sert à automatiser un peu plus l'obtention des mots à rechercher.

Historiquement dans l'entreprise, il y avait un fichier (fichier taxinomie) fait totalement à la main, où se trouvaient les fonctionnalités du produit avec des niveaux associés (niveaux 0 et 1).

Voici un exemple du fichier que l'on utilisait:

level 0:	Prix	Composition	Composition	Composition
level 1:	Prix	Reaction Allergique	Formulation	reactions nocives
	coût	allergie	Dimethicone	neurotoxique
	prix	reaction	huile	dépression
	budget	intolérance	alcool	léthargie
	dollars	allergène	hexylene	cancer
	dolar	provoquer	jojoba	Benzène

extrait d'un fichier taxinomie

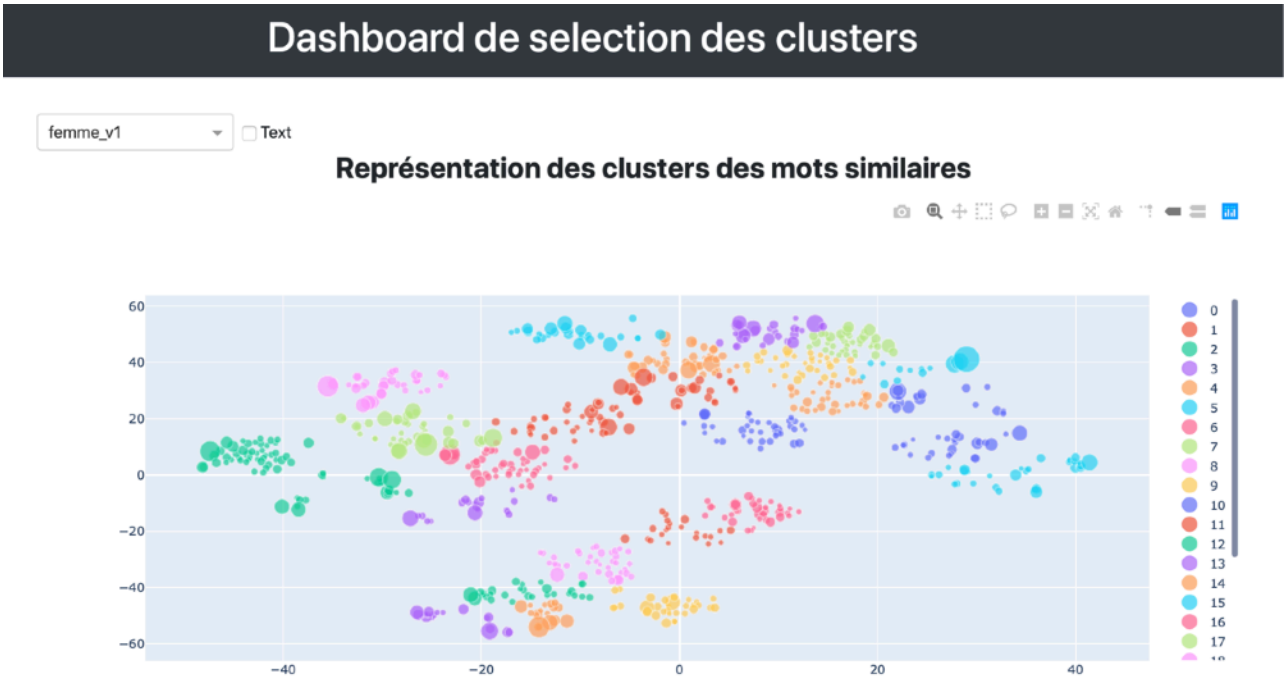
Maintenant on passe par un clustering de tous les mots sortant du préprocessing, accompagner par les notes olfactives du site fragrantica. On va aussi construire des trigrammes.

Pour avoir un bon clustering on va utiliser:

- TFIDF pour limiter les mots
- word2vec pour avoir les mots similaires
- kmeans pour avoir les clusters

J'ai beaucoup travaillé à ce moment plutôt sur le Dashboard, qui a était fait avec dashplotly. Ce Dashboard se compose:

- d'un accès sécurisé
- d'une visualisation de tous les clusters
- la possibilité de changer entre les produits hommes et femmes
- de pouvoir zoomer sur un cluster
- pouvoir supprimer un ou plusieurs clusters à tout moment
- pouvoir retirer des mots d'un cluster
- avoir une heatmap sur la similarité des mots entre eux
- pouvoir exporter un fichier csv contenant tous les mots sectionnés avec leur cluster associé.



Une partie du Dashboard de sélection es clusters

Par la suite on a fait le déploiement sur Heroku. Pour que les clients puissent facilement y accéder et de manière sécurisée.

Les clients n'étant pas beaucoup techniques et ayant peur de faire des erreurs qui impacteraient les étapes suivantes du projet. Ils nous ont demandé de faire plusieurs points avec eux.

Dans ces points il a fallu:

- voir si ont garde ou pas les différents clusters
- voir si ont garde ou pas certains mots dans les clusters
- faire des regroupements de cluster
- nommé les clusters ce qui va être notre niveau 1
- nommé des groupes de clusters qui vont le niveau 0

Les clients ont notamment demandé qu'on refasse un clustering avec les mots qui ont été retirer des clusters, pour pouvoir en crée de nouveaux. Mais aussi de grossir certains clusters déjà pré-établis.

Dans le premier cas (avoir un clustering sur les mots retirés) il suffit de refaire tournée un clustering que sur les mots ayant été supprimé.

Dans le second cas (avoir de plus gros cluster), on utilise le word2vec("Cf. annexes page - 48") et on va chercher les mots les plus proches, tout en se limitant aux mots les plus proches avec un seuil.

Mon stage c'est terminer après avoir écrit ces 2 scripts. Mais j'ai tout de même de fini le projet car après la première phase de clustering, j'ai continué la suite du projet, avec les parties que vous allez voir ci-dessous.

iv. Obtenir les sentiments d'une phrase

Pour obtenir le sentiment d'une phrase on va utiliser plusieurs modèles à apprentissage supervisé .

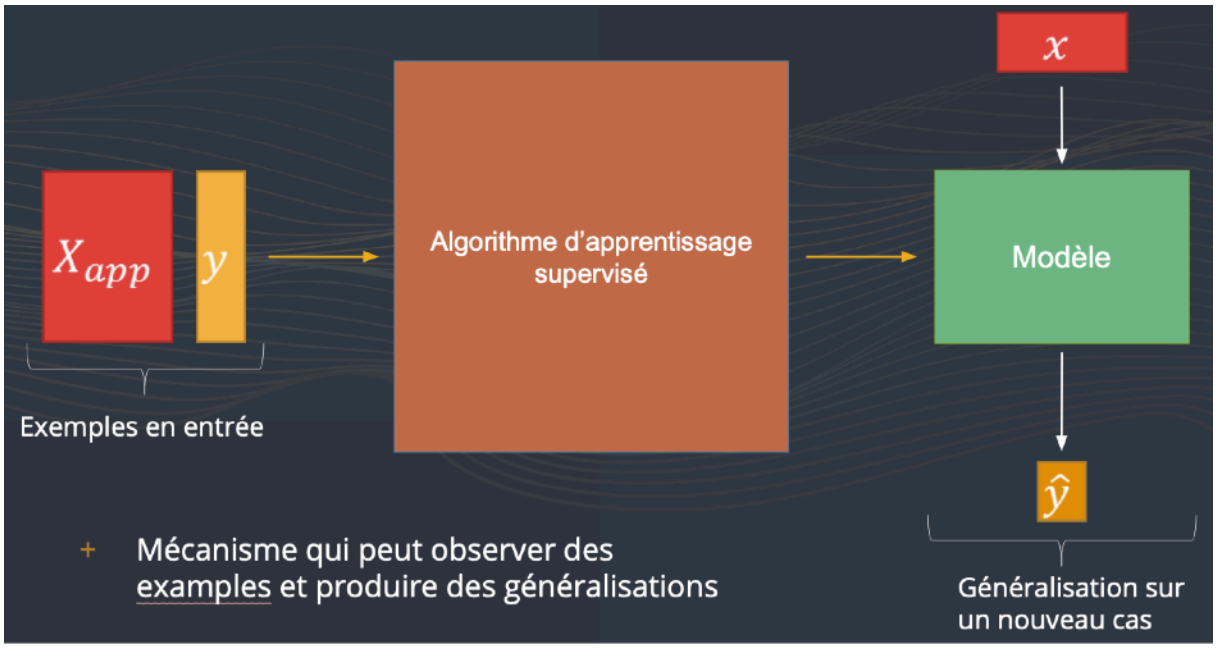


schéma explicatif d'un apprentissage supervisé

Avant de faire l'entraînement de ses modèles, il faut faire un minimum de préprocessing:

- mettre les caractères en minuscules
- retirer la ponctuation

On utilisera un TFIDF qui va nous donner l'importance des mots dans chaque phrase, grâce à sa fréquence d'utilisation.

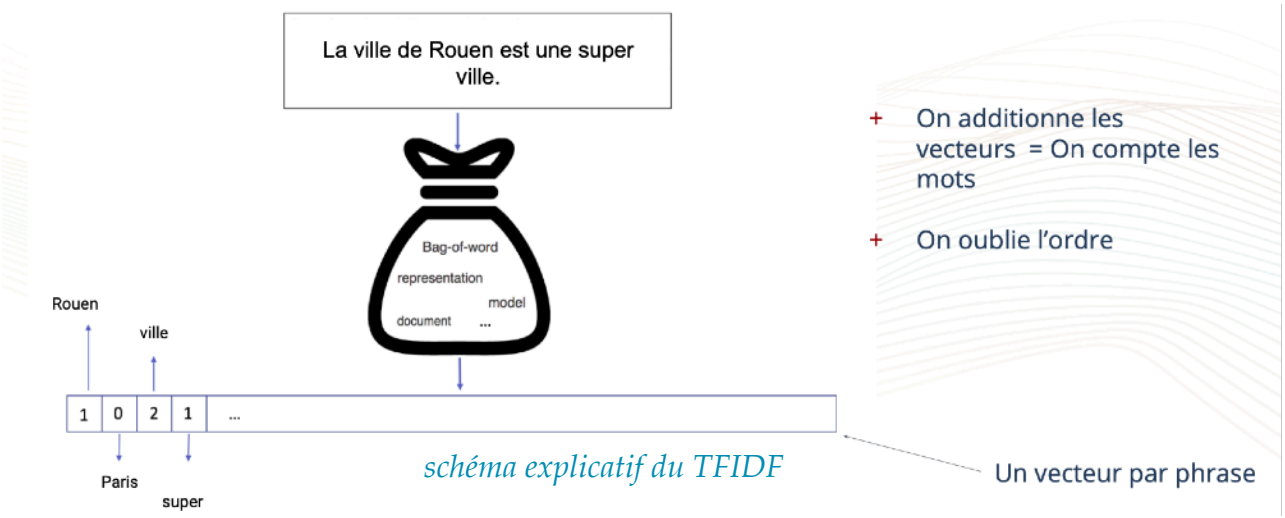


schéma explicatif du TFIDF

On va ensuite créer à partir de notre jeu de données deux autres jeux de données:

- un pour l'entraînement
- l'autre pour tester la fiabilité du modèle

On va bien faire attention en divisant notre jeu de données (dataset) à bien garder un échantillonnage stratifié (on va donc bien garder la même distribution de données avec un sentiment positif ou négatif dans les deux nouveaux jeux de données).

On peut donc maintenant entraîner nos modèles avec le dataset dédié à l'entraînement.

Dans le dataset initial on doit avoir :

- une feature (caractéristique) notre Review
- un label (étiquette) le sentiment positif ou négatif

La question est maintenant de savoir comment remplir le label associé a notre features.

stratégie pour obtenir le dataset initial

	Positif	Négatif
scénario 1	L'utilisateur laisse au moins 4 étoiles	L'utilisateur laisse moins de 4 étoiles
scénario 2	Prendre le complément de commentaire du site Beauté-Test « Point Positif »	Prendre le complément de commentaire du site Beauté-Test « Point Négatif »

Le scénario 1 était celui de base du projet, on n'arrivait pas à avoir de bonnes performances, car le dataset était déséquilibré. Le dataset reste toujours déséquilibré, même si on change le seuil du nombre d'étoiles ou encore en rajoutant une catégorie neutre (pouvant être prédit par certains modèles utilisés).

Le scénario 2 nous a données de très bonnes performances dès son implémentation. Pour avoir un maximum de données, on a fusionné nos Reviews avec d'autres Review Beauté-Test parlant maquillage (provenant d'un ancien projet), pour créer un plus gros dataset

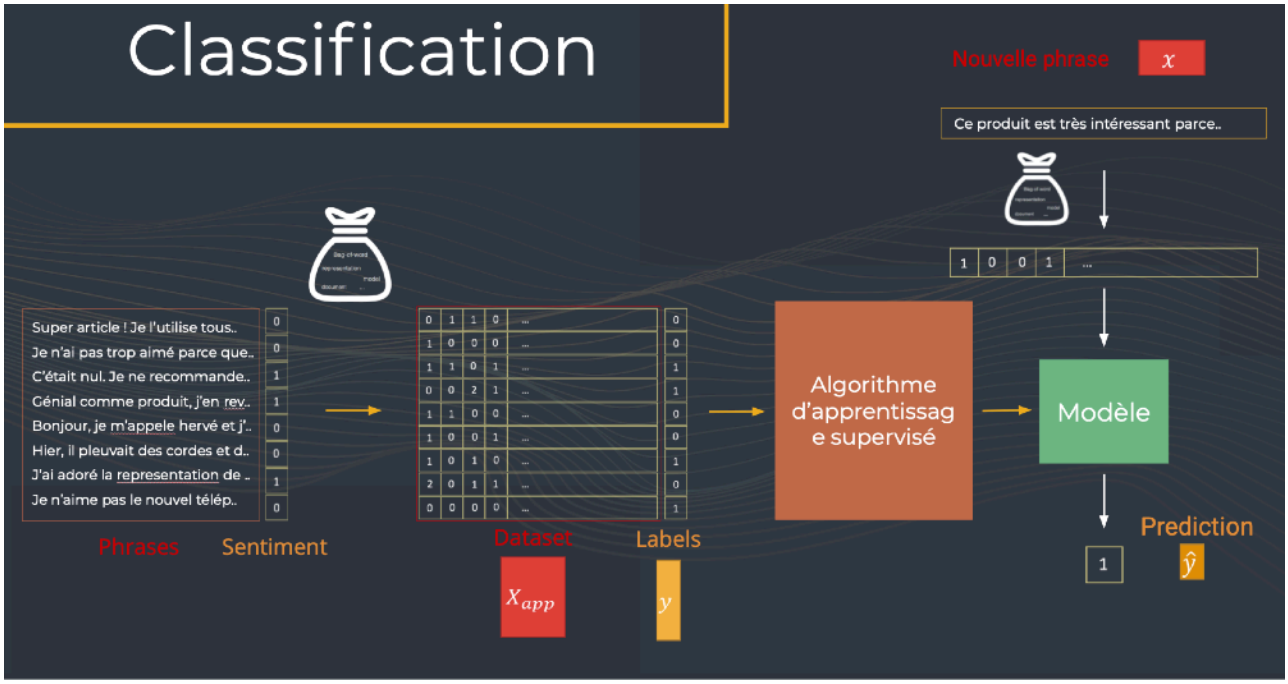


schéma explicatif de l'algorithme

On va par la suite faire une fonction d'inférence.

Cette fonction va avoir pour objectif d'utiliser les méthodes, ensemblistes. Ici, nous utilisons une moyenne, qui va être suffisante dans notre cas pour avoir de bons résultats.

Il faudra par la suite valider notre modèle pour cela il existe plusieurs méthodes comme:

- la courbe de ROC
- la matrice de confusion

Vous pouvez voir ci-dessus la matrice de confusion produite avec le dataset de test, en utilisant la seconde méthode. On voit qu'on a de très bonnes performances, par cette belle diagonale.

Matrice de confusion après inférence

matrice de confusion		matrice de confusion normalisé	
17652	958	95 %	5 %
557	17203	3 %	97 %

v. Construction d'une citation

La construction de citation est la finalité de ce projet.

Une citation va donc être une séquence (Review découpé au préprocessing) où l'on a trouvé une fonctionnalité du produit (feature), ou une opinion sur le produit, et le sentiment dans cette séquence.

On va donc dans cette partie réunir ce qui était fait précédemment, à savoir :

- préprocessing pour avoir les séquences
- fuzzy matching pour avoir un produit de référence
- recherche de sentiment

On doit aussi pouvoir chercher une opinion à partir d'une feature et vis versa.

Je vous rappelle qu'historiquement, on utilisait pour avoir les features le fichier « taxinomie », qui répertoriait que des features et pas des opinions. On devait donc chercher s'il y avait une feature dans notre phrase, par la suite, on cherchait les opinions associées à cette feature.

Maintenant avec le topic modeling on a des features, et des opinions. Il faudra donc adapter le script, si on veut faire que la recherche d'opinion à partir feature et vis versa.

J'ai donc eu à implémenter la cherche de feature à partir d'une opinion. J'ai profité pour rendre le code qui passait de feature à opinion plus lisible.

Cela m'a permis de comprendre en détail le code, mais aussi de ne pas avoir à tout réécrire pour implémenter ma partie, car la plupart du code est similaire. J'évite donc de la duplication de code.

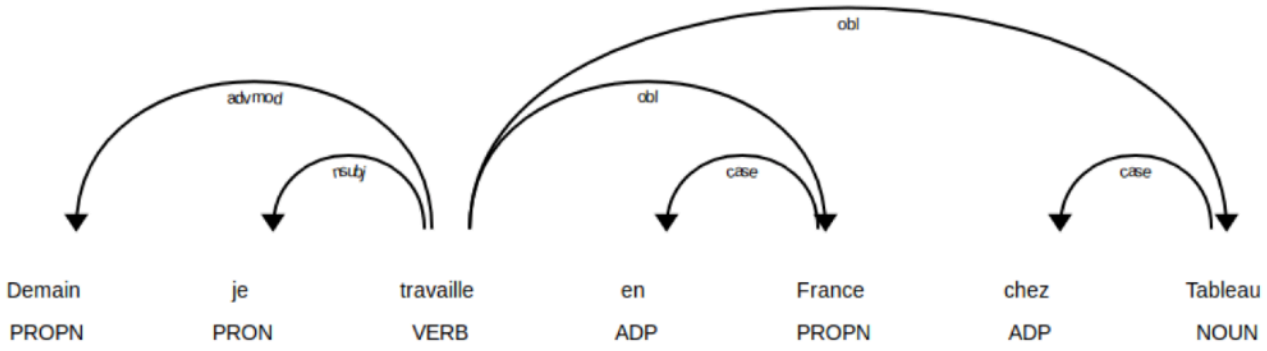
Voici comment se déroule la recherche d'une opinion à partir d'une feature. On va commencer par associer à chaque review leur produit de référence, trouver avec le fuzzy matching.

On va chercher si un topic (provenant du topic modeling) est dans une séquence. On va par la suite savoir si ce topic est une feature ou non en regardant son type grammatical (verbe, nom…) grâce à la lémmatisation faite au moment du préprocessing. Notre nouvelle feature sera trouvé en prenant le numéro du token associé pour avoir le mot de

base dans la séquence associé. On a donc maintenant une feature associer à notre séquence.

Par la suite, on va chercher dans notre séquence à partir de notre feature une opinion. Pour cela, on va utiliser le NER « SpaCy fr ». On va donc chercher si notre feature n'as pas d'enfant ou de parent ayant un type grammatical(adjectif, adverbe…) définie comme opinion.

Comme opinion peut-être composée de plusieurs mots, on va donc aller chercher les parents et les enfants de cette opinion et faire la concaténation de manière intelligente pour essayer le plus possible de garde le sens de l'opinion.



exemple de phrase dans SpaCy (avec disSpaCy)

Pour passer d'une opinion à une feature il faudra juste faire l'opération inverse.

Après avoir obtenu les informations sur les features et les opinions, il va falloir utiliser notre fonction d'inférence, pour obtenir le sentiment de la séquence. Une fois que le sentiment est trouvé on a toutes les informations pour construire la citation et la mettre en base.

Ces citations sont la finalité de ce projet. Car elles nous permettent par la suite de construire une application web, qui répertorie toutes les citations liées à un produit recherché. Et de pouvoir faire des statistiques sur les différentes catégories de ce parfum.

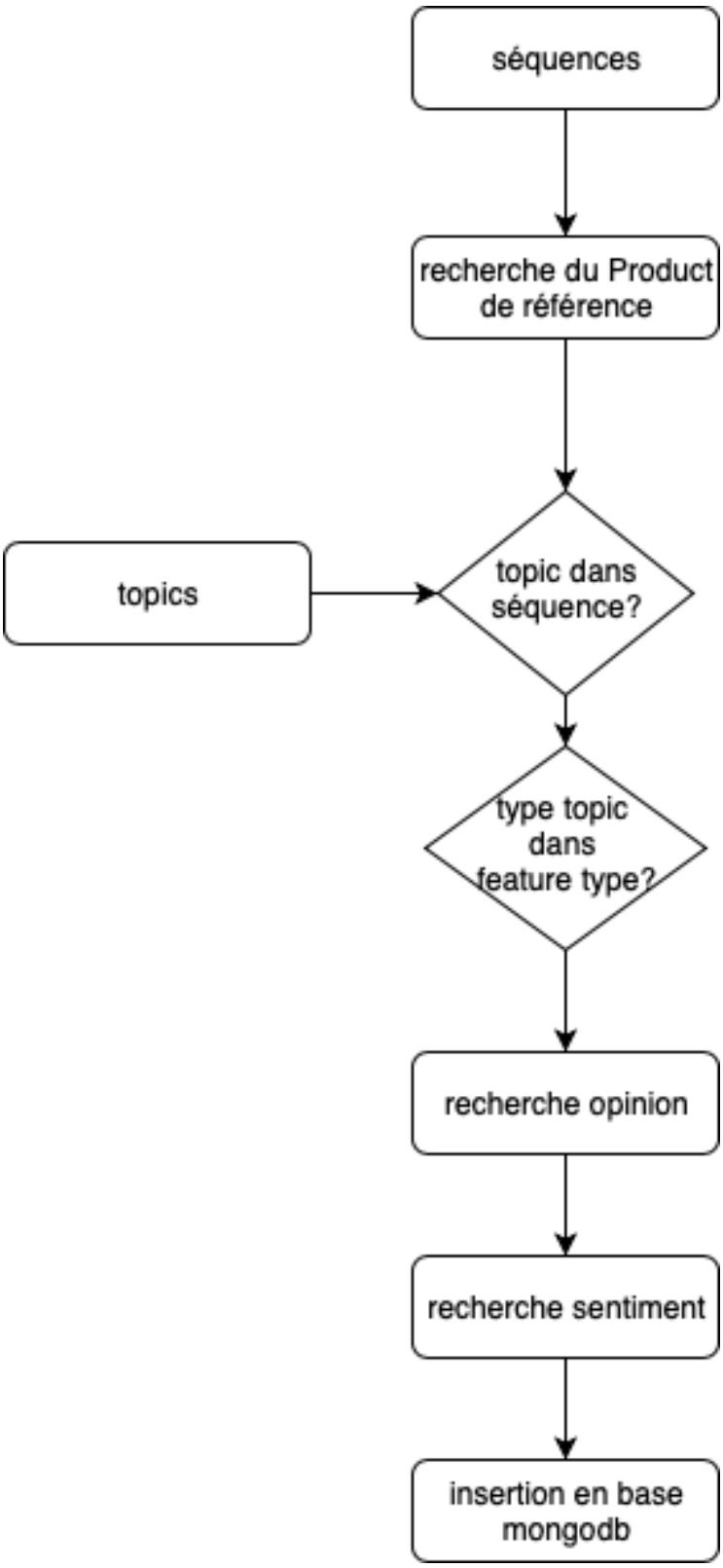
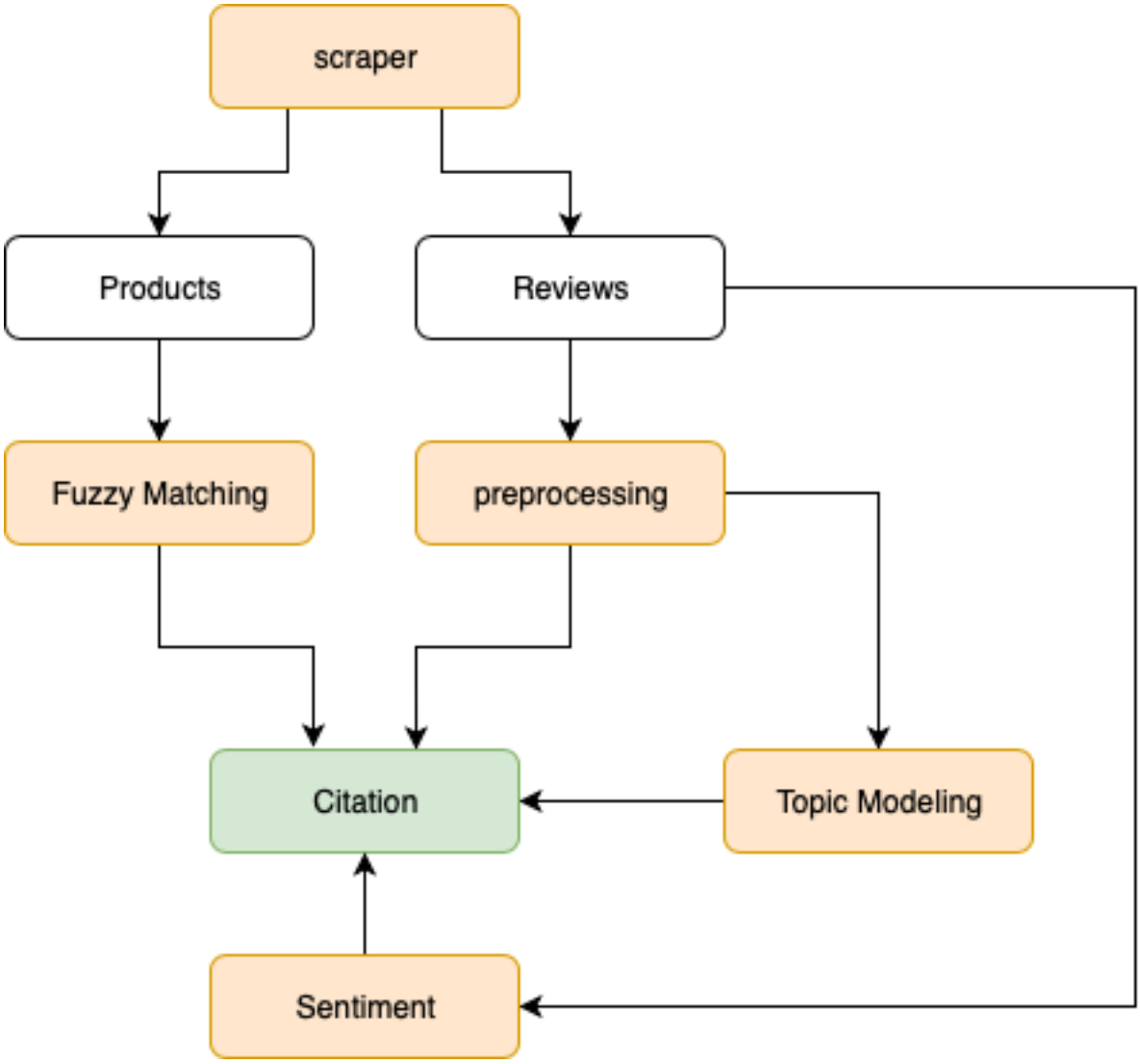


diagramme descriptif du processus de la création de citation



Pipeline du projet track_market

1. Activité secondaire

i. Déploiement d'application avec docker

L'entreprise a commencé à faire une migration de ses services des serveurs OVH vers le cloud AWS. J'ai aidé à faire sa migration, en réinstallant des applications comme :

- la base de données MongoDB
- Elastic Search, Logshtash et Kibana (ELK)
- liée ELK et la base de données MongoDB avec MongStash (pour garder les logs)
- mise en ligne d'un serveur FTP, qui met tous les fichiers dans un bucket S3

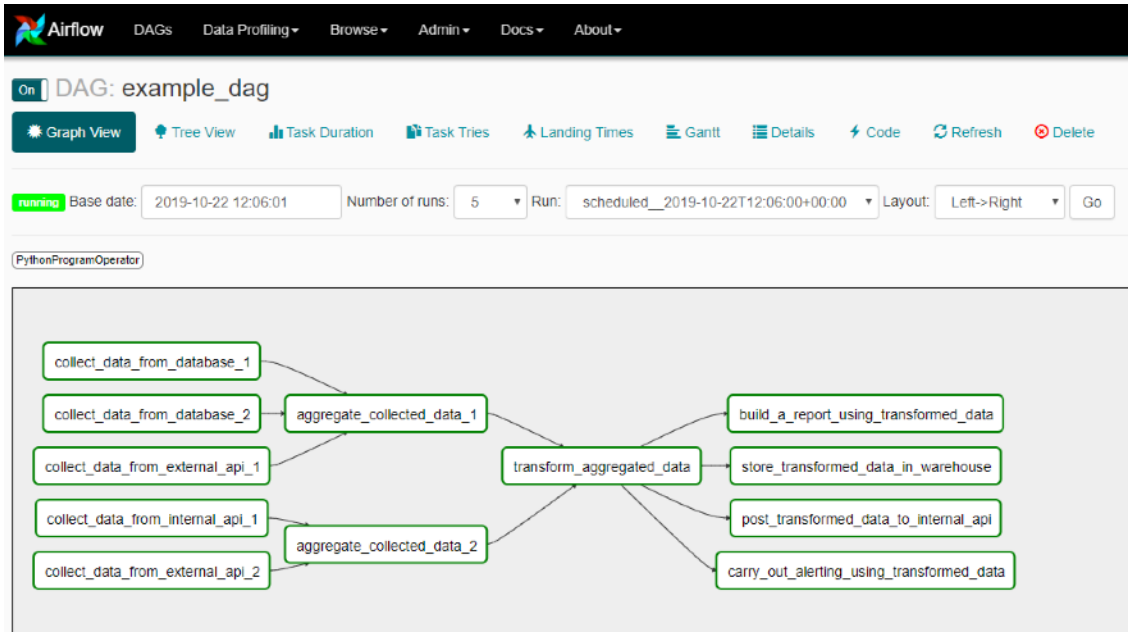
Pour cela, j'ai dû apprendre à utiliser à déployer des containers docker, et docker-compose.

ii. mise en production d'un algorithme de Machine Learning

J'ai par la suite dû refaire du Docker, mais plus en profondeur, pour aider au déploiement d'un autre projet de Machine Learning, où il fallait utiliser Airflow ("Cf. Annexe page - 49"), pour pouvoir programmer les différentes tâches à effectuer.

Airflow est constitué de plusieurs éléments, mais 2 sont importantes :

- les DAGs (Directed Acyclic Graph) qui peuvent être programmées ou déclenchés



exemple de DAG airflow

- les tasks(taches) qui constitue les DAGs (sont des fonctions python ou script Shell)

Le projet ici consistait à écouter les fichiers entrant sur un SFTP(stockage dans S3), ces fichiers étaient déposés par le client.

Il y a 2 types de fichiers :

- les fichiers qui vont lancer un apprentissage du modèle
- les fichiers qui vont lancer la prédiction du dernier modèle créé

Je me suis beaucoup occupé lors de ce projet dans airflow :

- de l'écoute de l'arrivée de ces nouveaux fichiers
- de la création dynamique des DAGs par rapport aux fichiers qui a été déposé
- du téléchargement et de l'envoi des fichiers entre S3 et la machine de déploiement avec boto3 ("Cf. Annexe page - 49")
- de l'insertion de certaines données en base mongoDB

On a commencé airflow, en faisant l'installation directement sur la machine. Hors de base airflow ne fait pas de multiprocessing entre les différentes tasks. Pour faire du multiprocessing, il faut utiliser celery qui a une installation assez complexe (nécessite Redis, une base PostgresSQL, flower…). Il a donc été naturel pour moi d'aller chercher si quelqu'un n'avait pas fait container docker, avec un docker-compose qui réglerait mon problème. Il y a bien quelqu'un qui a créé une image docker de ce type (https://github.com/puckel/docker-airflow).

Avec cette image docker, on a eu des problèmes d'environnement python. J'ai donc décidé de construire ma propre image docker héritant de l'image trouvée pour installer différents packages sur python qui ont une installation un peu spécifique. On a donc un docker-compose servant à déployer airflow et gérant le multiprocessing.

Pour la gestion des fichiers sur S3, j'ai créé un wrapper sur boto3 permettant:

- download des fichiers ou dossiers
- upload des fichiers ou dossiers
- retirer des fichiers ou dossiers de S3
- de voir les fichiers disponibles dans un dossier

Ce wrapper[17] gère aussi le globbing[18]. Il pourra être utilisé dans tous les projets de Factonics.

J'ai beaucoup travaillé aussi dans ce projet à l'élaboration de comment créer les DAGs et les tasks associés.

Je devais aussi permettre le versionning des modèles. Pour cela, nous avons utilisé S3 où l'on fait le stockage des déférents modèles en utilisant un format particulier. À terme une autre solution de versionning de ses modèles serait d'utiliser MLFlow.

c. Interprétation et critique des résultats

Sur mon sujet principal, la plus grosse parti aura été le scraping. Avec les différents problèmes que j'ai eus surtout au niveau d'Amazon. Ou j'ai dû réécrire tout le scraper, inclus les proxy…

Mais cette partie reste essentielle, car sans ces données, il est difficile d'avoir de bons résultats. Car il faut une masse de données assez importante pour avoir de bons retours sur un produit.

Le fuzzy-matching va nous permettre d'avoir les produits identiques. Les résultats obtenus sont très bons, comparés à au début. Néanmoins, il peut être encore amélioré par des méthodes bien spécifiques à certains cas. Mon maître de stage m'a dit que les résultats que j'avais obtenus était plus que satisfaisants pour notre cas et qu'améliorer ces résultats, pourraient être un sujet de stage.

Le préprocessing est une étape importante quand on fait du NLP. Le script a dû grandement changer, car on l'utilise aussi pour faire le Topic Modeling. Le Topic Modeling demande d'avoir le moins de mot différent, la lémmatisation n'était donc plus suffisante. En ajoutant la correction orthographique. J'ai eu de bien meilleurs résultats. Aujourd'hui, la seule amélioration possible serait de changer le lemmatizer SpaCy qui n'est pas toujours fiable.

Le Topic Modeling a été un nouvel élément du projet, il a remplacé une étape faite complètement à la main (le fichier de taxinomie). Cette étape pourrait être améliorée en utilisant un TFIDF entraîné sur les mots de Wikipédia.

Chercher les paramètres pour avoir un bon clustering reste quand même assez compliqué. Sur la partie de Dashboard, beaucoup d'amélioration aurait pu être faite pour

faciliter la création et validation des clusters. Il reste quand même un bon outil pour la visualisation des clusters.

Pour obtenir les sentiments, la méthode est bonne. Le modèle a même été utilisé dans un autre projet trackmarket qui fonctionne très bien.

La phase finale consistant à trouver les citations a été validé par mon maître de stage. On obtient des performances satisfaisantes. Néanmoins, il reste des améliorations pouvant être effectué comme :

- de changer le NER (passé de SpaCy à flair)
- faire de la récursivité pour trouver de meilleures opinions
- pouvoir faire la recherche de citations avec des n-grams déjà construites par le topic modeling
- pouvoir faire la recherche citations en sélectionnant le genre produit associé

Sur ce projet, j'ai vu toutes les étapes pour construire un algorithme de Machine Learning. Chaque point de ce projet est améliorable, je pense, mais le code d'aujourd'hui s'est amélioré par rapport à ce qu'il y avait au début autant au niveau fonctionnalité que propreté du code. Pour ce projet j'ai beaucoup:

- rendu le code plus lisible et explicite
- fait évoluer le code en code plus générique
- ajouter de la documentation

J'ai pu aussi lors de ce projet rencontrer les clients (par appel téléphonique, mais aussi dans leurs locaux), à qui servirait ce projet. Il a été bien de voir ce qu'eux pensaient qu'on faisait et ce qu'ils attendaient du projet. Il est aussi souvent assez difficile de trouver les mots pour faire comprendre à une personne pas du tout technique ce que l'on fait.

Les projets secondaires étaient souvent liés au déploiement d'application. J'ai donc beaucoup appris à utiliser docker. J'ai maintenant les bases de docker et docker-compose.

J'ai passé tout le dernier mois de mon stage à travailler en équipe sur le déploiement d'un projet avec airflow qui est un très bon workflow. Airflow demande tout de même beaucoup de puissance de calcul, de base pour toute la gestion de ses DAGs. Le déploiement de ce projet a été terminé.

IV. Conclusion Général

Factonics a été pour moi le stage idéal. J'ai eu un accueil chaleureux dans l'entreprise dès le premier jour, il y régnait une bonne ambiance de travail. L'entreprise a mis en place des moments de détente lors d'événements particuliers comme (la galette de rois, un pot de départ…). Factonics est une entreprise conciliante, qui a permis pendant les grèves d'effectuer nos activités en télé-travail.

Je n'ai donc pas regretté d'avoir fait un stage « long » de par l'ambiance et le travail toujours été intéressant. Mes attentes concernant ce stage, on était validé. J'ai pu apprendre le machine Learning et confirmé que cela me plaît bien. J'ai pu aussi apprendre à déployer des algorithmes ainsi que des applications.

Le projet qui m'a été attribué est pour ma part terminé, il est encore d'actualité dans l'entreprise malgré mon départ étant donné que le contrat n'est pas encore clôturé.

Pour ce projet, j'ai fait beaucoup d'optimisation, j'ai aussi créé beaucoup d'outils pour aider soit à la validation des différentes parties, ou encore pour aider à l'implémentation de nouvelle fonctionnalité (comme le scraper, les règles de fuzzy matching…).

J'ai pu aider à déployer en production l'algorithme de machine Learning développer par un autre stagiaire d'EPITA en utilisant Airflow, et fais du versionning en utilisant les buckets S3.

Il a fallu garder en tête tout au long du stage que nos algorithmes sont destinés à être un jour sur la plateforme en cours de développement dans l'entreprise. Ils doivent donc être déployables facilement.

Ce stage m'a donc permis d'acquérir de l'expérience dans le domaine de la data science, mais aussi dans l'organisation de travail (seul comme en groupe).

V. Bibliographie & Glossaire

[1] Python et le Machine Learning:

https://docs.google.com/spreadsheets/d/1eNBLcKqCVN9zZQvfGUmm5bAzsETqB_ugVOlUtmvJGYU/edit#gid=910850616

[2] Site web de Factonics:

http://www.factonics.com

[3] Le rapport de stage de Souleymane DIEYE et Imed SAOUD.

[4] Crawling and scraping:

Un crawler va chercher a indexer différente page d'un site internet.

Le web scraping est une technique d'extraction du contenu de site internet, via des script ou programme, dans le but de le transformer pour permettre son utilisation dans un autre contexte.

[5] Framework:

L'objectif d'un framework est généralement de simplifier le travail des développeurs informatiques, en offrant une architecture « prête à l'emploi » et qui leur permette de ne pas repartir de zéro à chaque nouveau projet.

[6] Natural language processing:

Le Natural language processing ou en français traitement automatique du langage naturel est un domaine multidisciplinaire impliquant la linguistique, l'informatique et l'intelligence artificielle, qui vise à créer des outils de traitement de la langue naturelle pour diverses applications.

https://www.ekino.com/articles/introduction-nlp-partie-i

[7] NLP:

Natural language processing (voir plus haut)

[8] Lemmatisation:

La lemmatisation désigne un traitement lexical apporté à un texte en vue de son analyse. Ce traitement consiste à appliquer aux occurrences des lexèmes sujets à flexion (en français, verbes, substantifs, adjectifs) un codage renvoyant à leur entrée lexicale commune ("forme canonique" enregistrée dans les dictionnaires de la langue, le plus couramment), que l'on désigne sous le terme de lemme.

[9] NER:

La reconnaissance d'entités nommées ou Named-entity recognition(NER) est une sous-tâche de l'activité d'extraction d'information dans des corpus documentaires. Elle consiste à rechercher des objets textuels (c'est-à-dire un mot, ou un groupe de mots) catégorisables dans des classes telles que noms de personnes, noms d'organisations ou d'entreprises, noms de lieux, quantités, distances, valeurs, dates, etc.

[10] Régression (statistiques):

La régression est un ensemble de méthodes statistique très utilisées pour analyser la relation d'une variable par rapport à une ou plusieurs autres.

[11] Matrice de confusion:

Une Confusion Matrix (matrice de confusion) est un outil permettant de mesurer les performances d'un modèle de Machine Learning en vérifiant notamment à quelle fréquence ses prédictions sont exactes par rapport à la réalité dans des problèmes de classification.

[12] courbe de ROC:

Une courbe ROC (receviez operating characteristic) est un graphique représentant les performances d'un modèle de classification pour tous les seuils de classification. Cette courbe trace le taux de vrais positifs en fonction du taux de faux positifs.

https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=fr

[13] Tor:

Tor est un réseau informatique superposé mondial et décentralisé. Il se compose d'un certain nombre de serveurs, appelés noeud du réseau et dont la liste est publique. Ce réseau permet d'anonymes l'origine de connexions TCP.

[14] Fuzzy Matching:

La recherche approximative ou recherche floue (fuzzy search en anglais) est le problème qui consiste à trouver des chaînes de caractères qui correspondent à un motif approximatif plutôt qu'à une correspondance exacte.

[15] Apprentissage supervisé:

L'apprentissage supervisé est une tâche d'apprentissage automatique consistant à apprendre une fonction de prédiction à partir d'exemples annotés, au contraire de l'apprentissage non supervisé.

[16] Echantillonnage stratifié:

Méthode ou on divise la population en groupes homogènes (appelés strates), qui sont mutuellement exclusifs, puis on sélectionne à partir de chaque strate des échantillons indépendants.

[17] Wrapper:

En informatique, une fonction wrapper est un programme dont la fonction principale est d'appeler une autre fonction.

[18] Globbing:

En programmation informatique, les modèles de glob spécifient des ensembles de noms de fichiers avec des caractères génériques. Par exemple, la commande Unix Bash Shell mv *.txt textiles/ déplace (mv) tous les fichiers dont le nom se termine par .txt du répertoire courant vers le répertoire textiles.

[19] MLFlow:

MLflow est une plateforme open source permettant de gérer le cycle de vie du ML, y compris l'expérimentation, la reproductibilité et le déploiement.

https://mlflow.org

VI. Annexe

Sommaire des Annexes

d. Documentation sur l'Entreprise

Extrait du rapport de stage de Souleymane DIEYE :

I. Introduction

1. Overview of the company

FACTONICS is a startup incubated by Centralesupelec founded in 2016 by Charles Dadi whose premises are located in Paris in the 9th district.

Firmly focused on the data economy, Factonics aims on the one hand to improve the exploitation of their customers' internal data through the implementation of applications. On the other hand, it offers new sources of value by innovatively exploiting analytical technologies. The end goal of the start-up is to democratize access to the learning machine.

One of the tools initially created with this in mind is the "Top-model" application, "Top Model" is a catalogue of "ready-to-use" algorithms proposed mainly to the fashion sector.

On this basis, Charles Dadi decided to develop an application on the same concept but this time open to all business sectors always with the objective of opening access to AI for all. Thus was born "Factonics App"

With just over 2 years on the clock and a Marketplace of algorithms (Top-Model) awarded with a Blue Ocean Award, Factonics quickly became a part of the industry, yet rather saturated with applications manipulating artificial intelligence.

How? By addressing a new target of non-technical profiles for which the I.A was not accessible without a heavy implementation cost.

Through the platform, the objective is to provide key-in-hand algorithms useful to business teams to allow the simplest possible use of AI regardless of the field (health, banking, cosmetics, marketing, etc.)

These days, Factonics has 7 employees and 3 interns, there are a lot of data profiles such as Data Scientist, Data Engineer, Machine Learner, Account manager, Juriste, Data Protector, Developers and Designer.

The team is pioneering on AI and data science topics. The startup combines the operational excellence and seriousness of a strategy consulting firm with the agility and technical quality of a tech startup.

It is divided into 3 divisions, the main one is data, which is responsible for all R&D processes around the algorithms feeding the platform, this pole is in close

relationship with the development pole which is responsible for maintenance, the addition of new functionalities as well as the integration of new algorithms.

The management and customer relations division complete this organization. It deals with the internal and external life of the company by ensuring publicity, setting up a conference, newsletter, medium article, call client, commercial proposal, solicitation, etc.

Tutored by the founder Charles Dadi, graduate of the Ecole Centrale Paris in machine learning and the Université Paris VII in random modelling. He began his career at Natixis as a quantitative strategist. He then joined Ekimetrics, a data-centric strategy consulting firm, as the lead scientific data. He also worked in fashion and luxury before founding Factonics.

Charles is dedicated to helping organizations adopt large-scale artificial intelligence to radically improve their performance and is deeply convinced that every project is motivated by the need to create real impact and quantitative for a client's business.

He regularly participates in conferences on AI and works as scientific co-director of the Master's Degree in Executive Data Strategy at Mediaschool exec.

Huy Dan
Responsable Data Engineer

Charles Dadi
Machine Learner & fondateur

Eugénie
Account Manager & responsable management

Xinze
Data Scientist

Rebecca
Responsable Juriste Data Protection

e. Documentation sur le matériel/les logiciels:

1. Dans toutes mes taches:

Python 3.7.5

Python est un langage de programmation interprété, multi-paradigme et multiplateformes.

https://www.python.org

Connexion ssh

Secure Shell (SSH) un protocole de communication sécurisé permettant d'utiliser d'utiliser une machine a distance.

Jupyter notebook

Jupyter est une application web utilisée pour programmer dans plus de 40 langages de programmation. Jupyter est une évolution du projet IPython. Jupyter permet de réaliser des notebooks, c'est-à-dire des programmes contenant à la fois du texte en markdown et du code. Ces notebooks sont utilisés en data science pour explorer et analyser des données.

https://jupyter.org

screen (GNU)

Screen (GNU Screen) est un « multiplexeur de terminaux » permettant d'ouvrir plusieurs terminaux dans une même console, de passer de l'un à l'autre et de les récupérer plus tard.

https://doc.ubuntu-fr.org/screen

pipenv

Pipenv est le programme pour gérer les paquets dans les projets Python. Il n'est pas installé de base avec Python et nécessite donc d'être installé avant de pouvoir être utilisé.

https://github.com/pypa/pipenv

re

Ce module fournit des opérations sur les expressions rationnelles.

https://docs.python.org/fr/3.6/library/re.html

Argparse

Argparse est le module d'analyse de ligne de commande recommandé dans la bibliothèque standard de Python.

https://docs.python.org/3/library/argparse.html

pandas

Pandas est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles.

https://pandas.pydata.org

tqdm

Ajoute des bars de progression.

https://github.com/tqdm/tqdm

itertools

implémente de nouveau itérateur

https://docs.python.org/2/library/itertools.html

2. Pour le scraping:

Requests

Requests est une librairie HTTP, écrite en Python.

https://fr.python-requests.org/en/latest/

Beautifulsoup

Est une Library qui va analyser des fichiers XML ou HTML.

www.crummy.com/software/BeautifulSoup/bs4/doc/

Scrapy

Est un framework de scraping.

scrapy.org

MongoDB

Est une base de donnée No-Sql orienté document, ne nécessitant pas de schéma prédéfini des données.

www.mongodb.com

PyMongo

PyMongo est une distribution Python contenant des outils pour travailler avec MongoDB, et est la façon recommandée de travailler avec MongoDB de Python.

https://api.mongodb.com/python/current/

Voluptuous

Est une Library python qui sert a faire de la validation de donnée.

Utilisé souvent avec pymongo.

https://github.com/alecthomas/voluptuous

3. Pour le NLP et le machine learning:

SpaCy

SpaCy est une Library de Natural Language Processing.

https://spacy.io

FastTest

FastText est une bibliothèque qui permet aux utilisateurs d'apprendre les représentations de texte et les classificateurs de texte.

https://fasttext.cc

scikit-learn

Library de machine learning python

https://scikit-learn.org/

Nltk

Natural language tool kit est une Library de NLP

https://www.nltk.org

Word2vec

Word2vec est un groupe de modèles utilisé pour le plongement lexical (word embedding). Ces modèles ont été développés par une équipe de recherche chez Google sous la direction de Tomas Mikolov.

Ce sont des réseaux de neurones artificiels à deux couches entraînés pour reconstruire le contexte linguistique des mots.

gensim

Library contenant des outils pour du NLP comme le word2vec
https://radimrehurek.com/gensim/

TSNE

L'algorithme t-SNE (t-distributed stochastic neighbor embedding) est une technique de réduction de dimension pour la visualisation de donnée. Il s'agit d'une méthode non-linéaire permettant de représenter un ensemble de points d'un espace à grande dimension dans un espace de deux ou trois dimension, les données peuvent ensuite être visualisées avec un nuage de points.

Kmeans

Le partitionnement en k-moyennes (ou k-means en anglais) est une méthode de partitionnement de données et un problème d'optimisation combinatoire. Étant donnés des points et un entier k, le problème est de diviser les points en k groupes, souvent appelés clusters, de façon à minimiser une certaine fonction. On considère la distance d'un point à la moyenne des points de son cluster ; la fonction à minimiser est la somme des carrés de ces distances.

TFIDF

le TF-IDF (de l'anglais term frequency-inverse document frequency) est une méthode de pondération souvent utilisée en recherche d'information et en particulier dans la fouille de textes. Cette mesure statistique permet d'évaluer l'importance d'un terme contenu dans un document, relativement à une collection ou un corpus. Le poids augmente proportionnellement au nombre d'occurrences du mot dans le document. Il varie également en fonction de la fréquence du mot dans le corpus. Des variantes de la formule

originale sont souvent utilisées dans des moteurs de recherche pour apprécier la pertinence d'un document en fonction des critères de recherche de l'utilisateur.

4. Pour le déploiement du Dashboard:

Dashplotly

Permet de crée des Dashboard facilement.

https://plot.ly/dash/

Heroku

Heroku est une entreprise créant des logiciels pour serveur qui permettent le déploiement d'applications web.

www.heroku.com

5. Pour le projet secondaire:

Docker

Docker est une plate-forme logicielle qui vous permet de concevoir, tester et déployer des applications rapidement. Docker intègre les logiciels dans des unités normalisées appelées conteneurs, qui rassemblent tous les éléments nécessaires à leur fonctionnement, dont les bibliothèques, les outils système, le code et l'environnement d'exécution.

https://www.docker.com

Airflow

Apache Airflow est une plate-forme de gestion de flux(workflow) de travail en code source libre. Elle a été lancée à la Airbnb en octobre 2014 en tant que solution pour gérer les flux de travail de plus en plus complexes de l'entreprise.

https://airflow.apache.org

Boto3

Boto est un package Python qui fournit des interfaces à AWS.

https://boto3.amazonaws.com/v1/documentation/api/latest/index.html

f. Documentation produite lors du stage

1. documentation principal

Trackmarket

[TOC]

Pipefile

Pour installer pipenv:

```
pip install pipenv
```

Pour installer les dépendance d'un PipFile:

```
pipenv install
```

Pour entrer dans l'environement pipenv:

```
pipenv shell
```

il ajoute le nom du projet pipenv dans le promp du shell

Pour lancer un script dans l'environement pipenv:

```
pipenv run python yourfile.py
```

ou

```
pipenv shell #se met dans shell
python yourfile.py #lance le script
```

Pour installer un package dans l'environement

```
pipenv install package
```

Pour plus d'informations:

```
pipenv --help
```

our sur le github de pipenv:pipenv

Connection Tor (ConnectionManager)

Pour installer et utiliser Tor, lire leREADME.md contenue dans FactonicsUtils/Connect/ConnectionManager

2. partie de la documentation d’installation de TOR

Install requirement

ubuntu

you can run to install tor on ubuntu

```
$ ./tor.sh install
```

to restart tor you can run

```
$ ./tor.sh restart
```

to stop tor you can run

```
$ ./tor.sh stop
```

Add server IP to autorized firewall IP list in the database server

```
$ sudo ufw allow from IP to any port 27017
```

python library required

```
$ pip install -r requirements.txt
```

install tor and piovxy:

ubuntu command

```
$ sudo apt-get install tor privoxy
```

arch-linux command

```
$ sudo pacman -Syu tor privoxy
```

update torrc

```
$ sudo sh -c 'echo "ControlPort 9051" >> /etc/tor/torrc'
$ sudo sh -c 'echo "HashedControlPassword $(tor --quiet --hash-password  factonics2017)"
```

Crawling and scraping

Les fichier pour le Crawling et le Scraping sont contenur dans:Collect/Scraping Pour plus d'information sur comment faire du Crawling et Scraping lisez ce README.md

Amazon scraping

Les fichiers pour le scraping de Amazon sont dans:Collect/Scraping/amazon_scraping/amazon

Vous pouvez aussi lire README.md. Pour savoir comment utiliser le Scrapy, et ainsi scraper Amazon

Beaute-Test scraping

Les fichiers pour le scraping de Beaute-Test sont dans:Collect/Scraping/amazon_scraping/BeauteTest Le scraper le plus récent est scrapeBeauteTest2.py Pour en savoir plus sur ce scraper lancer le avec `--help` Attention a avoir bien installer TOR sur votre machine car ce scraper l'utilise

Fragrantica scraping

Les fichiers pour le scraping de Beaute-Test sont dans:Collect/Scraping/Fragrantica Le scrapeur à utiliser scrapeFragrantica.py Attention ce scraper est tres long, car Fragrantica bloque l'ip pendant un moment. Il faut mieur le faire tournée sur plusieur machine avec les flags `--num`, `--tot` Pour plus d'information lancer le script avec `--help`

Marionnaud scraping

Les fichiers pour le scraping de Marionnaud sont dans:Collect/Scraping/Fragrantica Le scrapeur à utiliser scrapeFragrantica.py

Sephora scraping

Les fichiers pour le scraping de Beaute-Test sont dans:Collect/Scraping/Fragrantica Le scraper le plus récent est scrapeSephora2.py

Fuzzy matching

Ce script a pour but de faire une correspondante entre les produits d'une sources (ici les produits de Fragrantica) sur d'autre sources (les autre produits).

Pour configurer les Fuzzy matching il faut modifier les parametres dans le__init__.py et fichier config.py. Le fuzzy matching se base sur la collection "products" pour pouvoir matcher les fichiers, et va mettre dans la collection "edges" la sortie On peut voir les différentes règles, pouvant être appliqué dans le fichier Matching.py. Pour lancer le fuzzy matching il suffit de lancer avec python fuzzyMatching.py.

Enrichment

Load Data:

Le script, va charger toutes les "reviews" dans un fichier ".pkl". Et va ajouter "point fort" et "point faible" des reviews de Beauté Test par defaut le path du fichier ".pkl" va etre dans la variable PROCESSEDDATAPATH qui est définie dans Application/NLPEnrichment/**init**.py

Preprocessing:

Ce script fait le preprocessing des reviews contenue dans un ".pkl" par default il utilise le path PROCESSEDDATAPATH

Topic Modeling:

Il est composer de plusieur script se trouvant dans:Application/NLPEnrichment/topicModeling.

Le script topicModeling.py un fichier .json ou est stocker la config. Il va esuite pouvoir crée différents clusters, avec les mots contenue dans le fichier préprocess donnée. attention d'avoir un fichier de config bien formaté, vous pouvez retrouver des fichiers de config ici Ce script a aussi un flog `--deep_clustering` qui a recrée des cluster avec un fichier de topic modeling ou on a retirer les lignes.

Le script enlargeClusters.py va servir comme son nom l'indique à élargir les clusters, qui utilise plusieurs flag, utiliser `--help` pour en savoir plus.

Load Taxinomy:

deprecated

Ce script a été replacé par le script deTopic Modeling

Sentiment Training:

Le script va train les models servant à l'inférence, pour savoir le sentiment de la phrase (si il est positif ou non). Pour cela il y a 2 sénario: - On utilise les étoiles des reviews - On utilise les "Point Fort" et "Point faible" contenue sur le site de beaute-test (méilleur pref avec ce scénario)

Sentiment Inference:

Ce script la load tout les models train dans lescript de Sentiment Triningt va faire l'inférence sur une liste de phrase. Cette inférence est la moyenne des probabilité retournée par les différent model.

Matching Citation:

Ce script va chercher les citations et les features dans les différente séquence.