Data Science en R/Python au Centre de Services des Données

NOM Prénom :
FRANK Shir

Branche :
ISI - VDC

Responsable Pédagogique :
MATTA Nada

Semestre :
Printemps 2023

Résumé (150 mots)

Mon stage s'est déroulé au sein de la Deutsche Bundesbank en tant que Data Scientist dans le centre de service des données.
Mes missions ont consisté à analyser et exploiter de gros volumes de données avec Python et R afin d'en ressortir une valeur spécifique, qu'elle concerne l'optimisation du workflow ou l'augmentation de la qualité des données. La collecte de données faisait aussi partie intégrante du stage. Les différents projets concernaient les tâches suivantes, réalisées de manière agile :
- Crawling des résultats de recherche Google - Comparaison de gros jeux de données et optimisation - Analyses de gros jeux de données - Génération automatisée d'un rapport de données - Travail avec ChatGPT et optimisation de l'input
L'enjeu est de valoriser au maximum les données du service en en générant de nouvelles qui soient compréhensibles et utilisables par les différents corps de métier présents à la Bundesbank.

Entreprise : Deutsche Bundesbank

Lieu : ***Mainzer Landstraße 16, 60325 Francfort-sur-le-Main, Allemagne***

Responsable : Stefan BENDER

Mots clés (cf Thésaurus) :

- Mise en place, mise en oeuvre
- Services des organismes financiers
- Informatique
- Analyse des données - Logiciels

utt
UNIVERSITÉ DE TECHNOLOGIE
TROYES

DEUTSCHE
BUNDESBANK
EUROSYSTEM

Remerciements

Je tiens tout d'abord à remercier Yvan, le responsable de l'association, ainsi que l'équipe RH de Gaea21 pour m'avoir recrutée au sein de l'association, et en particulier Charlotte Boll, ma coach RH, pour sa gentillesse, sa bonne humeur et son perfectionnisme.

Merci aussi à Alexandre Nguyen et Jonathan Capitao, mes tuteurs, pour leur aide quotidienne dans les tâches que je devais réaliser.

Sans oublier mon équipe de développeurs du Répertoire Vert, avec qui la cohésion était présente et qui étaient motivés pour atteindre nos objectifs, et mes autres collègues toujours là pour aider.

Table des matières

Introduction

La Deutsche Bundesbank est la banque centrale de la République fédérale d'Allemagne. Son rôle principal est de maintenir la stabilité des prix et la valeur de l'euro. Elle est également chargée de superviser les banques et les institutions financières en Allemagne, d'assurer la sécurité et l'efficacité des paiements et des opérations financières, de gérer les réserves de change du pays, et de participer à la formulation et à la mise en œuvre de la politique monétaire européenne en tant que membre de l'Eurosystème.

Elle est donc importante au niveau national, mais également international :
- Elle est membre de la Banque des règlements internationaux (BRI) et participe activement aux forums internationaux tels que le G20 et le Fonds monétaire international (FMI).
- Au niveau européen, elle est un membre important de la Banque centrale européenne (BCE) et joue un rôle clé dans la mise en œuvre de la politique monétaire de la zone euro.

La banque compte 30 filiales, comptant la centrale et les antennes régionales. Ces dernières mettent en œuvre les missions de la banque centrale allemande au niveau régional et sont responsables de la coordination avec les acteurs économiques locaux et les autorités régionales pour favoriser le développement économique régional.
Mon stage s'est déroulé à la Bundesbank Zentrale à Francfort-sur-le-Main, le siège principal de la banque. Il est le centre décisionnel et opérationnel de la banque et coordonne les activités des filiales régionales.

Je travaille dans le département DSZ (Datenservicezentrum) faisant partie du département Statistique, et dirigé par Stefan Bender. Son rôle est de fournir un accès aux données statistiques produites par la Bundesbank, ainsi qu'à d'autres sources de données pertinentes pour la politique monétaire et la supervision bancaire.
Plus spécifiquement, la DSZ offre les services suivants :

- Collecte et traitement des données provenant de différentes sources (banques centrales, agences gouvernementales et organisations internationales)
- Diffusion des données sous forme de tableaux, graphiques, rapports et bases de données interactives, accessibles aux décideurs politiques, chercheurs et analystes.

Il est composé de plusieurs sous-unités :

- DSZ 10 et 20 pour la recherche qui utilisent les données collectées par les autres sous-unités
- DSZ 30 chargé des tâches de traitement de données avec Machine-Learning, Data Engineering et la mise en place de la plateforme de données interne
- DSZ 40 pour la collecte de données
- DSZ 1 (Sustainable Finance Data Hub) travaille en étroite collaboration avec la DSZ 40

Plus précisément dans la DSZ, je fais donc partie de l'équipe du Sustainable Finance Data Hub (SFDH), une unité spécialisée qui collecte et enrichit les données liées au climat pour soutenir la prise de décision optimale en matière de changement climatique. Il fournit

un point d'accès central aux données, répond aux questions méthodologiques et prend part à des projets de génération de données innovants.

I Sujet et place dans le service

A Sujet

A.1 Sujet défini avant mon arrivée

Avant le début de mon stage, il était prévu que je travaille sur un projet innovant visant à développer une preuve de concept pour extraire des informations numériques sur les émissions de carbone à partir de rapports de durabilité en pdf. Pour ce faire, nous devions utiliser des technique de Natural Language Processing en python.

Le soutien aux tâches en cours afin de connaître toutes les fonctions clés de l'équipe faisait également partie intégrante du stage. Cela aurait pu inclure la documentation des données, la liaison et la comparaison des ensembles de données pour la gestion de la qualité des données et la participation aux groupes de travail internes et externes.

A.2 Sujet réel

Mes tâches en réalité étaient plus variées et sur divers projets :

- Crawling de résultats de recherche Google pour récupérer les rapports de durabilité des entreprises, et extraction d'informations des PDF
- Optimisation de code Python pour la comparaison de gros jeux de données
- Rapprochement de jeux de données provenant de différentes sources avec R (analyses)
- Génération automatisée d'un rapport à partir de résultats stockés dans un fichier csv
- Projet de recherche NLP pour trouver quels datasets et méthodologies sont utilisé(e)s dans quels papiers de recherche

B Fonction occupée dans le service

Ma fonction dans le SFDH, et plus généralement dans la DSZ était celle de stagiaire Data Scientist, aux côtés d'un autre stagiaire occupant exactement la même fonction. Comme détaillé dans le sujet, des tâches diverses en rapport avec les données m'ont été assignées sur de nombreux projets.

Le point commun entre toutes mes tâches était qu'à partir d'un gros volume de données, je devais en créer de nouvelles qui soient utilisables pour le but souhaité et compréhensibles par les différents corps de métier présents à la Bundesbank. Autrement dit, il s'agissait de tâches de valorisation des données. Pour cela, je passais par diverses méthodes de traitement de données à l'aide de Python ou R. J'ai parfois effectué le travail de collecte de données, notamment pour le projet GAIA (crawling).

Les projets ne se restreignaient pas au SFDH, ils concernaient aussi (et pour la plupart) d'autres services de la DSZ, notamment ceux dans la recherche : Certaines tâches m'ont alors été attribuées non pas par mon tuteur, mais par d'autres collègues d'autres services.

II Explication des projets

A Projet Gaia

Contexte

Le projet GAIA vise à simplifier l'utilisation des rapports de durabilité des entreprises grâce à une plateforme centralisée et à faciliter la recherche en réduisant les efforts manuels. Il cherche également à créer une référence fiable pour les informations liées au climat.

Ce projet a été lancé par la Deutsche Bundesbank en réponse à la prise de conscience croissante de l'importance de la finance durable. En effet, des risques financiers en découlent, et il faut donc en avoir conscience et savoir les gérer.

La motivation derrière le projet GAIA est de fournir aux investisseurs et aux décideurs des informations fiables et cohérentes sur la durabilité des investissements. Il s'agit de donner aux investisseurs les moyens de prendre des décisions d'investissement plus éclairées et de mieux comprendre les risques et les opportunités associés aux investissements durables.

La Bundesbank a également pour objectif de promouvoir la stabilité financière en intégrant les risques liés au changement climatique dans ses analyses et en encourageant les investisseurs à intégrer ces risques dans leur propre prise de décision.
L'équipe du projet est composée de membres de la Bundesbank bien sûr, mais se fait surtout en collaboration avec d'autres banques internationales (Banque d'Espagne, Banque Centrale Européenne, etc...).

Concrètement, le projet consiste tout d'abord à récupérer les rapports de durabilité des entreprises, disponibles publiquement en ligne. Ensuite, le texte et les tableaux / graphiques sont extraits, et les données d'empreinte carbone des entreprises (Scope 1, 2, 3) sont extraites à l'aide de Natural Language Processing (NLP).

FIGURE 1 – Mock-up de la plateforme Gaia

Antécédents

Mises à part les spécifications et objectifs de la plateforme qui étaient dejà clairs, rien n'avait été concrètement commencé avant mon arrivée.

Objectif visé

Le but est de présenter une preuve de concept du projet.

Mon principal objectif était donc de développer un crawler qui parcourerait les résultats de recherche Google pour les rapports de durabilité des entreprises majeures dans le monde. Les PDFs trouvés doivent ensuite être téléchargés. Parallèlement, les PDFs/liens trouvés doivent être classés, selon s'ils sont justes ou faux pour une requête donnée. Une fois que cela serait fait, je devrais présenter mes résultats à l'équipe du projet.

B Projet ESCB Exchange

Contexte

ESCB signifie Eurosystem/European System of Central Banks. Cela se réfère au système européen de banques centrales. L'Eurosystem est composé de la Banque centrale européenne (BCE) et des banques centrales nationales des pays de la zone euro.

L'ESCB Exchange est un forum permettant aux banques centrales de collaborer autour de données climatiques communes. Ce forum favorise l'échange de connaissances, de code et d'applications pour améliorer la valeur analytique des données climatiques utilisées. Chaque banque centrale désigne jusqu'à 2 experts en données climatiques pour participer au forum. Les rencontres du forum ont lieu chaque mois, et des groupes volontaires et agiles pilotent les sujets d'intérêt commun. L'initiative encourage l'échange sur le nettoyage des données, le partage de code et les leçons apprises concernant les données climatiques granulaires (des entreprises).

L'un des sujets abordés lors des forums est notamment le rapprochement de données sur les émissions carbones, censées être identiques mais provenant de différentes sources. En effet, les banques collectent ces données et les utilisent pour des recherches. Il est donc primordial de choisir la meilleure source de données climatiques. Ma mission portait exactement là-dessus.

Antécédents

Concernant le rapprochement de données provenant de différentes sources, un code d'analyses de données en R ainsi qu'une présentation présentant la comparaison avaient déjà été faits. La comparaison concernait les sources ISS et Carbon 4 Finance pour les données de 2020, tous deux fournisseurs de données.

Objectif visé

Ma mission consistait en la même analyse que celle faite précédemment, sauf qu'au lieu de faire une analyse bilatérale, je devais en faire une trilatérale en ajoutant aux sources précédentes celle de TRUCOST. Le Powerpoint associé devait également être fait en aval.

FIGURE 2 – Logos des sources de données à rapprocher

C Projet CSDB

Contexte

CSDB signifie "Central Securities Database". Il s'agit d'une base de données centrale des titres en Allemagne, utilisée pour la surveillance des marchés financiers, le suivi des transactions de titres et la facilitation des processus de règlement-livraison. Au sein de la DSZ, les chercheurs utilisent cette base de données.

Les datasets sont sous la forme de fichiers CSV et il y en a 1 pour chaque mois de chaque année depuis 2009. Tous les mois environ, une nouvelle version d'un ou plusieurs de ces datasets peut parfois être générée, avec des nouvelles colonnes, nouvelles valeurs, etc... Autrement dit, une nouvelle version du dataset de 2022-04 (différente de l'original) peut être créée en Avril 2023.

Pour chaque dataset, 4 variantes doivent être générées : En effet, des normes existent, et chaque groupe de chercheurs possède des droits de lecture différents sur les colonnes du dataset. Par conséquent, chaque variante possède le même contenu dans ses colonnes, mais les colonnes présentes sont différentes. Et cela doit être fait dès qu'une nouvelle version d'un dataset existe. Or, chaque dataset fait plusieurs Go, et actuellement, cette génération des 4 variantes ainsi que la comparaison de chaque nouveau dataset avec sa version originale prend 1 semaine.

Quand la procédure doit être répétée tous les mois, cette solution n'est pas viable telle quelle et doit être optimisée.

Antécédents

La situation à mon arrivée était la suivante :

Les nouvelles et les anciennes versions du "même" ensemble de données (par exemple les données 2009-04) étaient comparées après la génération des 4 ensembles de données. Cela engendre beaucoup d'effort et de temps de calcul évitable.

Les fichiers csv ont été compressés afin de tenter une baisse du temps, ce qui a effectivement beaucoup accéléré le temps de lecture des csv. Cependant le temps de compression semblait fortement ralentir le processus.

La comparaison des datasets comportait trop d'informations (moyenne, médiane de chaque colonne des datasets, et bien d'autres valeurs statistiques), ce qui utilisait très certainement du temps supplémentaire inutile, quand seulement un très bref résultat était attendu.

Objectif visé

Afin de pouvoir générer les 4 versions de datasets uniquement sur les versions modifiées des jeux de données de la même période, la comparaison doit être réalisée en amont. De plus, elle doit être optimisée et n'indiquer à la fin que si les datasets sont identiques ou non, et sinon montrer les différences. L'idée est aussi d'analyser les résultats de la méthode de comparaison choisie, par exemple en cas de valeur exacte mais format différent, ou en

cas de colonnes présentes que dans l'un des deux datasets.

D Projet NFiG

Contexte

NFiG signifie "pour usage interne seulement" (Nur für internen Gebrauch) et concerne les données des chercheurs du département. Ces derniers utilisent Stata, un logiciel de statistiques pour générer des graphiques ou effectuer différentes analyses. Lorsque ces graphiques doivent être publiés, ils doivent être adaptés de manière à ne plus être confidentiels.

Par conséquent, chaque graphique doit être retrouvé dans le code qui l'a généré, ce qui implique de connaître le fichier stata et la ligne à laquelle l'output est généré.

Le but est donc de retrouver automatiquement où est créé chaque graphe dans le code, ou de manière plus générale, de faciliter leur "traçage". Or, les exportations peuvent être générées par des boucles, ce qui rend la recherche des noms de fichiers difficile.

Antécédents

Un code python avait déjà été écrit pour tenter d'automatiser cette recherche, mais n'a pas vraiment réussi. Des rapports Excel on été faits à la main afin de recenser quel graphique ou autre fichier output a été généré dans quel fichier de code / fichier log et à quelle ligne.

Objectif visé

La 1ère tâche qui nous avait été donnée était une première lecture et compréhension du code déjà fait pour rechercher les citations de fichiers générés.

Puis la seconde consistait en la génération automatique d'un rapport en markdown grâce à Python, à partir des fichiers Excel remplis manuellement.

Plus de tâches auraient pu m'être données, mais nous avons préféré avec l'autre stagiaire nous concentrer sur d'autres projets. Il était en effet assez difficile de comprendre la personne en charge de ce projet lorsqu'une nouvelle étape devait être commencée, celle-ci expliquant de manière très abstraite sans montrer concrètement de quoi il s'agit. Nous avions de plus le choix de travailler sur bien d'autres projets aussi intéressants.

E Projet de recherche NLP

Contexte

Ce projet fait l'objet d'un papier de recherche appelé "Exploiter les grands modèles linguistiques pour extraire les citations de jeux de données et de méthodologies".

Il est utile pour les chercheurs de la Bundesbank de voir facilement quels datasets et méthodologies sont utilisés dans quels papiers de recherche déjà produits. En effet, il sera ensuite plus facile pour eux de déterminer quels datasets et méthodologies sont les plus adaptés pour leur recherche.

Concrètement cela se traduirait par un système de recommandation orienté utilisateur : "Les chercheurs comme vous ont utilisé tels datasets et telles méthodologies" ou encore "Vous avez utilisé tel dataset, celui-ci pourrait aussi vous intéresser".

Cela permettrait également d'identifier quels datasets ont débouché sur des résultats de qualité et de manière plus générale, quels datasets sont disponibles pour la communauté de recherche. Cela confère plus de valeur aux datasets, à savoir des informations de contexte : Comment les datasets sont-ils utilisés, quelles analyses permettent-ils, etc...

Cela peut aussi permettre d'améliorer les services de la DSZ, qui pourra ainsi proposer des jeux de données plus spécialement adaptés aux besoins des chercheurs.

Antécédents

Cette idée de recherche et sa valeur ajoutée, ainsi que les observations retirées, avaient déjà été documentées dans un livre écrit par les data scientists et chercheurs du département, portant sur les données dans les papiers de recherche. Les leçons tirées regroupent entre autres :

- Les datasets ayant un même nom dans le papier de recherche peuvent référer à des datasets en réalité différents
- Les datasets mentionnés peuvent se référer à des datasets utilisés dans les recherche, mais aussi simplement cités

Ces leçons soulignent la complexité du problème à résoudre.

L'idée est d'utiliser un Large Language Model (LLM) déjà existant de manière à ce qu'il nous donne directement la liste des Datasets / méthodologies mentionnées. Le LLM utilisé serait par exemple ChatGPT, pour des résultats optimaux. 2 schémas pour la potentielle procédure à suivre ont été faits par un collègue :

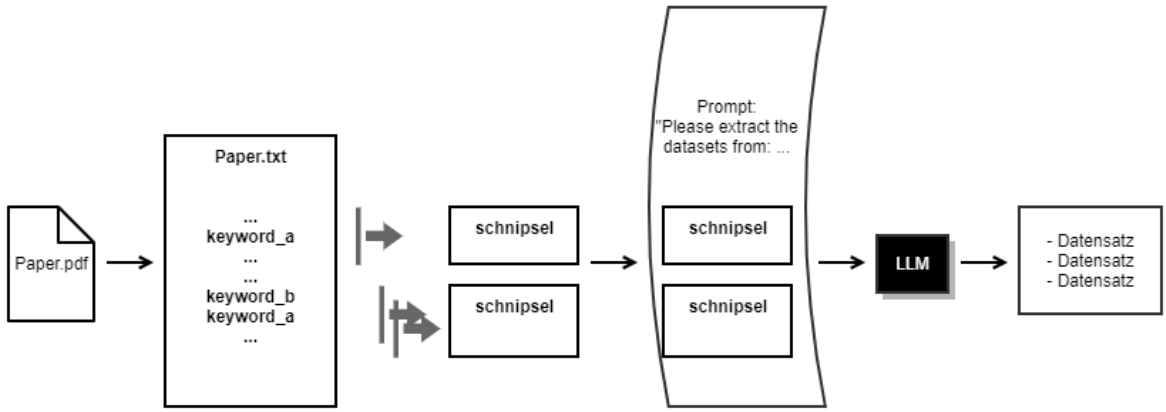

FIGURE 3 – Prototype A du projet de recherche

La 1ère idée serait donc de découper les papiers en différents paragraphes, et de demander à ChatGPT de directement donner la liste des Datasets / Méthodologies contenues dans chaque paragraphe.

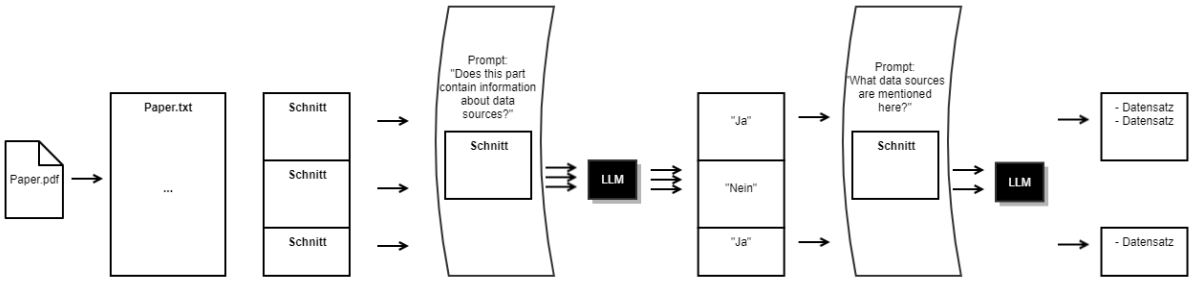

FIGURE 4 – Prototype B

Cette 2ème idée est un peu plus prudente et procède en 2 temps : 1- Demander si qqchose d'intéressant est contenu dans l'extrait 2- Si oui, demander quoi exactement.

Cela pourrait être répété soit avec chaque phrase individuelle, soit avec des extraits plus grands.

Objectif visé

L'objectif est donc à la fin de réaliser un papier de recherche faisant part de cette expérimentation. Il s'agit avant tout de savoir utiliser ChatGPT, de savoir lui donner le contexte adapté afin qu'il rende le résultat le plus juste possible.

Concrètement, il faudrait idéalement que Le LLM nous donne la liste exacte de tous les Datasets et Méthodologies utilisés, et qu'il sache reconnaître à quel dataset il est fait référence, même lorsqu'il n'est pas directement mentionné dans un certain extrait, mais qu'il l'a été précédemment. Ex : "La 1ère vague de données" -> Bonne réponse = la 1ere vague du Dataset X.

III Déroulement du travail

A Méthodologie et Organisation

Durant toute la durée du stage, il y avait un suivi régulier, permettant une gestion de projet selon la méthode agile.

Chaque jour, une réunion d'alignement de 30 min s'appelant "Jour Fixe" avec mon équipe (dont mon tuteur), le SFDH, avait lieu. Cela permettait à chaque membre de tenir les autres au courant de ce qu'il a fait la veille et de se faire aider par l'équipe en cas de questions.

Au début de chaque mois se déroulait additionnellement le Jour Fixe avec Stefan Bender, le responsable du département DSZ, afin d'aborder des points organisationnels et le tenir au courant du travail général de l'équipe du SFDH.

Un Jour Fixe DSZ se tenait tous les 3èmes mercredis du mois, avec l'ensemble du département, pour un alignement général des différents sous-départements.

Au milieu du stage, un "Feedback Call" avec mon tuteur était l'occasion de partager mon ressenti sur le stage, ce que je voulais faire pendant les mois restants, si les projets précédents m'avaient plu, etc... Cela permettait aussi à mon tuteur de me donner son avis sur mon travail, et d'éventuellement m'indiquer les points à améliorer.

Comme le travail avait lieu en hybride, toutes les réunions se faisaient en visio via Webex. J'ai d'ailleurs bénéficié d'un ordinateur portable afin de pouvoir faire du télétravail plusieurs jours par semaine. La communication avec les membres de la Bundesbank se faisait alors via la messagerie Jabber et par mail.

Au niveau de l'organisation concernant purement mes tâches, le travail m'était toujours donné tâche par tâche, ce qui me permettait de ne pas être débordée et d'être aussi plus efficace. De plus, cela encourage des réunions fréquentes et ainsi une gestion de projet agile.

Dès qu'une tâche était terminée, j'organisais une réunion d'avancement avec les personnes concernées pour avoir leur feedback ainsi que de nouvelles tâches.

Sécurité informatique

S'agissant d'une grande banque centrale, la sécurité est bien sûr prioritaire. Les ordinateurs ne peuvent être connectés à internet seulement lorsqu'ils sont connectés au réseau de la Banque (directement ou par VPN), et seuls certains sites sont autorisés.

La Banque possède un serveur interne dans lequel tous les dossiers et fichiers sont partagés. Chaque dossier est protégé et n'est accessible qu'à certains groupes.

Si un nouveau logiciel a besoin d'être installé, il doit être commandé via la plateforme Servity, puis l'installation doit être déclenchée par le service IT. De même pour certains accès, notamment pour Gitlab ou Python, qui sont à demander sur la plateforme BIAM.

La mise en place de Python comprend notamment la création d'environnements virtuels pour chaque projet, afin d'éviter des problèmes aux changements de version de Python.

Heureusement, tout cela est très bien guidé sur Confluence, une plateforme de documentation partagée, où sont notamment disponibles des guides pour la mise en place de Python sur les machines de la banque. Elle est également utilisée pour documenter certains projets avec l'état des lieux actuel, les recherches déjà effectuées, les étapes suivantes, etc...

B Application de la méthode et Résultats

B.1 Projet Gaia

La toute première étape du projet était de participer à une réunion avec Manuel Fangmann, un membre du pôle Innovation de la Bundesbank, qui participait activement au projet durant l'ensemble de mon stage. Le projet m'a donc été présenté et ma 1ère mission attribuée : Ecrire un code python qui parcourt les résultats Google avec comme entrée 'companyName + "sustainability reports" + year + "pdf"' et télécharger tous les pdf.

La liste d'entreprise est dans un premier temps celle du DAX40, qui réunit les 30 plus grandes entreprises cotées à la Bourse de Francfort. J'ai décidé de tester la recherche avec une liste d'années allant de 2017 à 2022.

Il m'a été demandé de travailler sur mon ordinateur personnel pour ce projet, car étant donné que l'accès à la quasi-totalité des sites est bloqué sur les machine de la banque, il serait compliqué de téléchargé les pdfs depuis ces sites.

En 2 jours, j'ai donc implémenté le Google Crawler à l'aide du package requests-html, et téléchargé les pdfs les plus pertinents, mais les résultats n'étaient pas des meilleurs. En effet, les résultats de recherche n'incluaient pas que des pdfs, et ceux trouvés n'étaient donc parfois pas les plus pertinents.

J'ai ensuite tenté de récupérer les titres des PDFs ou des liens tels qu'ils sont indiqués pour les utilisateurs afin de pouvoir y lire le nom de l'entreprise et l'année, mais je n'ai pas réussi et cette entreprise était dans tous les cas inutile, puisque ces informations pouvaient souvent se trouver directement dans le lien.

Dans ma recherche d'amélioration, j'ai trouvé un package donnant des résultats plus pertinents : Beautiful Soup. J'ai également mis en place un comptage des liens "douteux", c'est-à-dire ceux ne contenant ni le nom de la bonne entreprise, ni l'année. Les différents liens sont désormais triés à la volée selon s'ils sont douteux ou non, et ces 2 listes ensuite sauvegardés dans le fichier json correspondant.

Après la 1ère réunion d'avancement, j'ai mis en place beaucoup d'améliorations à mon code :

- Calcul du pourcentage de pdfs justes trouvés par an
- Amélioration du tri des pdfs trouvés/non trouvés : Désormais, un pdf est défini comme douteux s'il ne contient pas l'année ET/OU pas le nom de l'entreprise, et non pas uniquement si aucun des deux n'est présent.
- Ajout d'un opérateur de recherche avancée pour n'obtenir que les les liens qui menant à un pdf (filetype :pdf) (-> il n'y a donc plus de rapports "not found", seulement

des pdfs douteux)
- Parfois, seuls les 2 derniers chiffres de l'année sont notés dans le lien, ou encore l'une des deux parties du nom de la société (Ex : "Telekom" dans "Deutsche Telekom"), j'en ai donc tenu compte
- Un problème qui revenait assez fréquemment était que la bonne année pouvait être présente dans le lien, mais que dans le nom du fichier, une autre année était indiquée. Afin de passer en priorité l'année présente dans le nom du fichier, j'ai simplement vérifié que l'année était inscrite précisément à cet endroit, sans inspecter le reste du lien.
- les PDFs ne peuvent être définis comme trouvés uniquement si le lien contient le terme "Report"

La principale difficulté lors de la mise en place de ces améliorations était la gestion d'une condition très longue pour la définition d'un lien "non trouvé", qui est d'autant plus négative, avec à la fois des OR et AND. J'ai finalement simplifié le résonnement en faisant un condition positive pour les liens "trouvés".

Pour anticiper la suite du projet avec l'étape de Text Mining des PDFs, il m'a été demandé de tester l'extraction de textes et d'images des PDFs, ce que j'ai réussi sans difficulté avec le package Fitz.

Concernant le téléchargement des PDFs, j'avais des difficultés à tous les télécharger, certains sites bloquant les requêtes lorsqu'elles sont détectées comme étant automatisées. Cela se traduisait soit par une requête n'aboutissant jamais mais sans erreur, soit par un code de réponse "403-Forbidden". Un autre cas d'impossibilité du téléchargement est lorsque les rapports sont incorporés à la page web, avec un "viewer intégré", ou encore lorsque le pdf n'est plus disponible sur la page en question.

J'ai pu résoudre le cas de requête infinie grâce à une liste de "User Agents" qui seraient choisis à chaque fois aléatoirement pour être inclus dans le header de la requête. L'erreur 403 n'était en revanche pas solvable, même en tentant avec un Pool de Proxys.

Lors d'une réunion suivante, l'idée de lire les PDFs afin de mieux les classer a été suggérée. Je me suis donc attelée à cette tâche et ai extrait le texte des 3 premières pages des rapports dont le lien est douteux afin d'y chercher l'année, le nom de l'entreprise (ou partie du nom) et le terme "Sustainable Report" ou un dérivé. Le lien était donc reclassé en conséquence.

Cela a considérablement amélioré les résultats, car ne considérer que le lien était très réducteur, et beaucoup de rapports douteux étaient en réalité justes. Mais seules les 3 premières pages étaient optimales à lire, car après une autre année pouvait apparaître, bien qu'il s'agissait du rapport d'une autre année, faussant les résultats. Lire moins de 3 pages était en revanche trop peu pour trouver toutes les informations requises.

Cette lecture de PDFs n'a cependant pas résolu tous les problèmes, bien qu'ils soient rares :

- Le nom de l'entreprise ou même l'année ne sont parfois inscrits que sous forme d'image, non reconnue dans l'extraction de texte.

- L'année recherchée peut être écrite au début du rapport sans qu'il ne s'agisse du rapport de cette année-là. Celui-ci est donc noté comme trouvé alors qu'il est faux.

Après la prochaine réunion est venue l'étape de Refactoring du code, afin de 1-Rendre le code qui devenait de plus en plus fourni, plus lisible. 2- Rendre les différentes étapes indépendantes les unes des autres.

Mon code pouvait être découpé en 5 parties : Préparation de la liste d'entrperises, Crawling (=recherche des liens), Téléchargement des pdfs, Lecture des fichiers, et écriture des statistiques finales des résultats trouvés. Il y a donc 1 fichier par étape, en plus du Main qui les appelle toutes.

Pour que toutes ces étapes soient plus indépendantes les unes des autres, il fallait que le tri des liens se fasse en deux temps : une première fois en même temps que le crawling, et une deuxième fois après la lecture automatique des rapports douteux. J'ai donc 4 fichiers de résultats où se trouvent les différentes listes des liens obtenus pour chaque requête : found-list et doubt-list avant le téléchargement, et de même après le téléchargement. Le fichier found-list après téléchargement est composé de la copie de la 1ère found-list, mais avec en plus de nouveaux liens qui étaient au départ douteux.

La procédure est détaillée plus clairement ici :

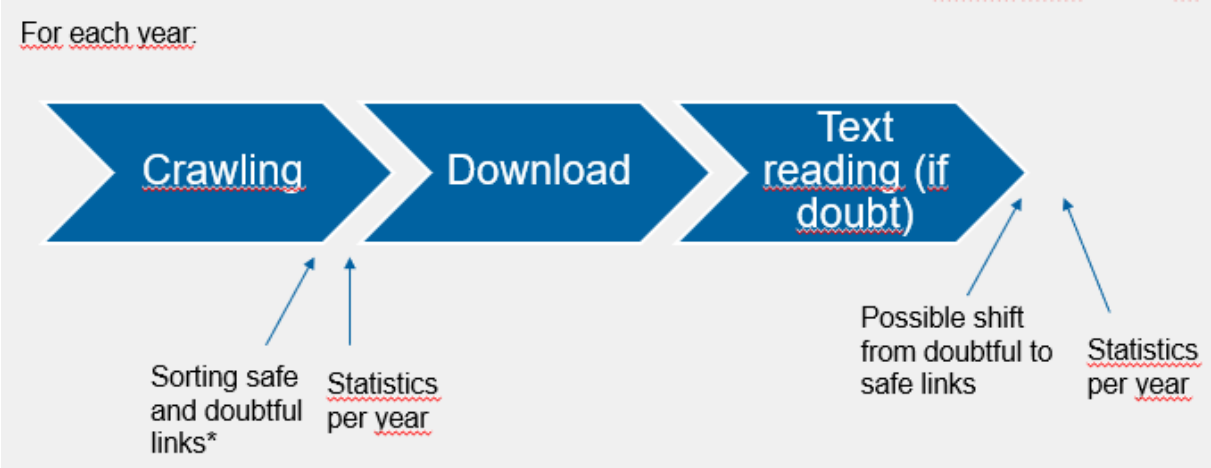

FIGURE 5 – Procédure du code

Une fois que tout fonctionnait bien, il était temps de tester le code sur un volume beaucoup plus grand d'entreprises, à savoir la liste de celles du MSCI World, Index boursier pour les plus grandes entreprises dans le monde entier. Cette liste en contient 3425 différentes. En ajoutant celles du DAX40, j'ai un échantillon de 3455 entreprises. Néanmoins, le fichier csv fourni contenait des doublons, et les noms des entreprises étaient parfois beaucoup trop longs (Ex : CONTEXTLOGIC INC CLASS A). J'ai donc dans un premier temps fait une liste des termes "interdits" et ai tronqué les noms à partir de ces termes pour rendre une liste de noms propre (notamment "inc", "group", "plc", "class", etc.).

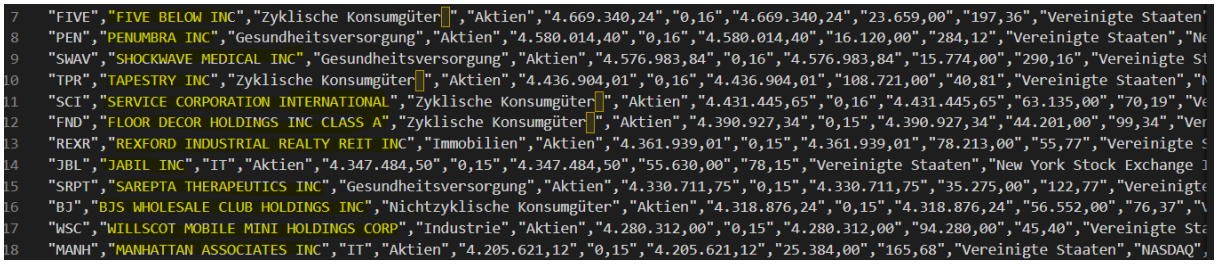

FIGURE 6 – Extrait de la liste MSCI avec les noms des entreprises en jaune

Pour ce projet, nous bénéficions d'une instance AWS Guacamole afin d'avoir une meilleure performance, le code étant très long à s'exécuter. Mais tout ne s'est pas passé comme prévu :

- Lorsque mon code s'exécute sur l'instance, il s'arrête au bout de 5min avec une Exception provoquée par une réponse vide de Beautiful Soup. Je devais donc redémarrer l'instance afin que cela remarche, mais le problème revenait toujours, à une entreprise différente. En local, cela se produisait aussi, mais beaucoup plus tard. J'ai essayé de trouver une solution en cherchant un autre moyen de crawler, et suis tombée sur l'API Google Custom Search, qui fonctionnait beaucoup mieux avec des résultats fiables, mais qui avait un nombre de requêtes maximal par jour très bon pour la version gratuite.
- Le blocage du téléchargement que j'avais résolu en local, apparaissait sur l'instance de nouveau, ce qui bloquait l'ensemble du code.

J'ai donc organisé un court meeting afin de mettre l'équipe au courant de mes problèmes et de demander de l'aide. Cette réunion m'a bien débloquée grâce aux solutions proposées : En réalité, lorsque le code s'exécute rapidement, Beautiful Soup était "débordé" et n'avait pas toujours le temps de donner une réponse. Avec un simple "time.sleep" de 0,8 secondes, le problème était résolu. La seule conséquence était que le temps d'exécution était un peu plus long.

Concernant le problème de téléchargement, j'ai mis en place un timeout et attrapé l'exception levée afin de continuer le code malgré tout. Les PDFs pour lesquels une exception a été levée sont sauvegardés dans un fichier afin de pouvoir les compter après-coup. J'ai donc implémenté le calcul du pourcentage par an et du nombre total de PDFs n'ayant pas pu être téléchargés.

Une journée de travail entière ne suffisant pas pour que le code s'exécute complètement, j'ai dû trouver une solution pour qu'il puisse reprendre là où il en etait lorsque l'instance se déconnecte ou que je dois fermer la page.

Parmi les différents changements à effectuer pour cela dans le programme, il fallait rendre les parties "web crawling" et "téléchargement/lecture des pdfs" complètement indépendantes l'une de l'autre. Lors de la 1ère étape, au lieu de remplir des tableaux qui se vident à chaque nouvelle relance, il était pertinent de directement remplir les fichiers json (doubt-results0 ou found-results0) regroupant tous les liens trouvés ou non trouvés (ces fichiers n'étaient auparavent remplis qu'à la fin de chaque année parcourue). Ainsi à la 2ème étape, ces fichiers json pouvaient être lus pour trouver les liens et non dans les tables

contenues dans les variables. Ces liens étaient ensuite placés dans l'un des 2 nouveaux fichiers (doubt-results1 ou found-results1), au niveau de l'année courante.

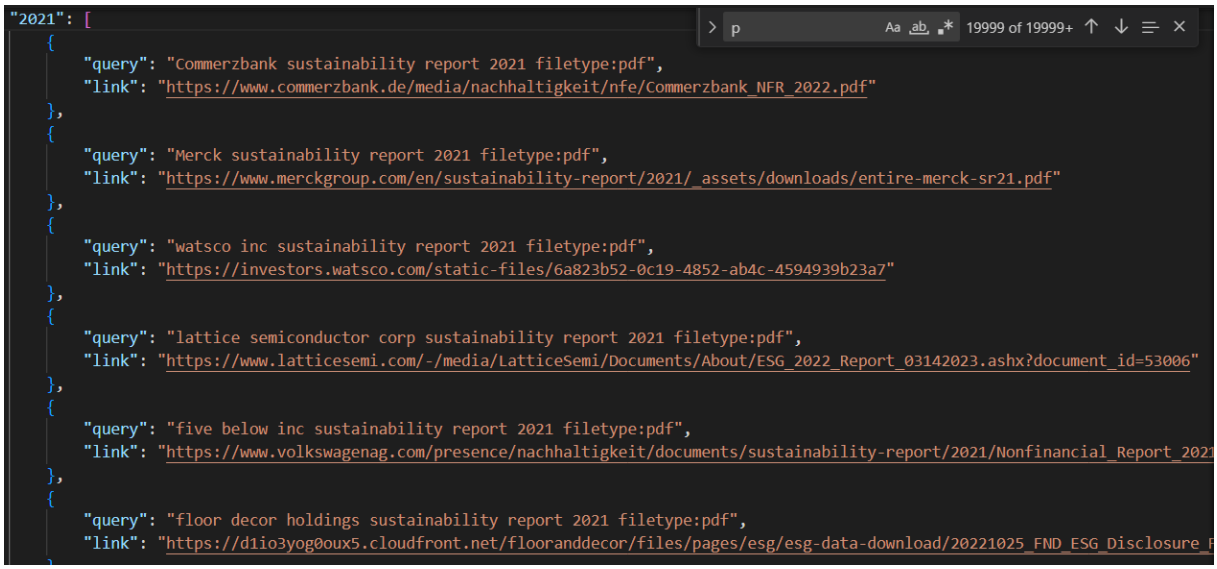

FIGURE 7 – Extrait des résultats douteux après la lecture des pdfs (doubt-results-1.json)

La dernière entreprise traitée avec l'année en cours était notée dans un fichier txt lu à chaque nouvelle relance.

En parcourant le fichier csv de la liste MSCI, j'ai remarqué que j'avais oublié beaucoup de termes interdits. Mais je devais à la fois vérifier que les rapports étaient toujours trouvés avec le nom de l'entreprise raccourci. J'ai donc passé un petit temps à nettoyer les noms. Un peu plus tard, j'ai remarqué que certaines entreprises avaient besoin des termes interdits pour être trouvées (Ex : Post holdings / software ag problem / fp corp). J'ai donc modifié ma stratégie et ai instauré une liste de mots tolérés, et une de mots interdits. Ainsi, tous les termes après les mots tolérés sont supprimés (Ex : Tous ceux apres "inc"). Pour que la recherche du nom dans les rapports fonctionne toujours, j'ai mis en place la même règle que pour les liens : que seule une partie du nom peut être présente.

J'ai donc pu tenter de lancer mon code pour de bon sur l'instance AWS, mais le fait d'y enregistrer les quelques 20 000 rapports dans le Repository Git posait problème, et plus aucune commande ne fonctionnait (git push, pull, add, status etc...). Même en tentant de mettre en place git LFS, rien n'y faisait. Afin de pouvoir accéder aux PDFs et étant donné que l'instance ne disposait que d'un terminal, j'ai pensé à uploader les PDFs directement sur Dropbox à la volée grâce à l'API, puisqu'ils devaient dans tous les cas y être à la fin. Cela fonctionnait sur mon PC, mais pas sur l'instance, j'ai donc abandonné l'idée de l'utiliser. L'exécution était donc plus pratique et même plus réussie sur ma machine personnelle, le téléchargement étant moins souvent bloqué par les sites.

Alors que j'avais bien entamé l'exécution en local, j'ai remarqué que toutes les heures environ, le token expirait. Il fallait donc manuellement en regénérer un nouveau. Il était donc finalement plus simple de d'enregistrer les PDFs en local et d'après coup tout uploader

en même temps. J'ai pu utiliser le forfait Business de Dropbox de mon tuteur pour y uploader les 136GB de PDFs.

Les dernières observations que j'ai faites sur mes résultats, sont que :

1) Les résultats contenaient des doublons, soit à cause du fichier MSCI initial, soit quand je relançais le code, et qu'une entreprise qui avait déja été faite avant, soit refaite une 2ème fois.

2) qu'en fin de compte, le fait de considérer comme juste un lien qui contient les 2 derniers chiffres de l'année recherchée, faussait les résultats plus qu'autre chose : certains liens contenaient juste une suite aléatoire de chiffres qui ne faisaient pas référence à l'année, et ils étaient considérés à tort comme justes.

J'ai donc supprimé cette condition, résolu le bug des doublons, et relancé le code une bonne fois pour toute, qui avait besoin de plus de 2 jours pour s'exécuter complètement.

Mes fichiers statisitques contenaient désormais plus d'informations qu'au début : Pourcentage de PDFs trouvés pour une année spécifiques, nombre cumulé de PDFs douteux, trouvés, et à trouver au total :

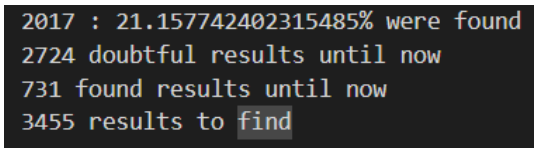

FIGURE 8 – Statistiques de 2017 avant le téléchargement des PDFs

Pour finir, je devais faire une présentation de quelques slides afin de présenter mes résultats à l'équipe Gaia. Un léger refactoring du code (division en plus de fonctions) et surtout sa documentation étaient également nécessaires.
Les résultats obtenus pour ce projet sont donc présentés sur les slides suivantes :

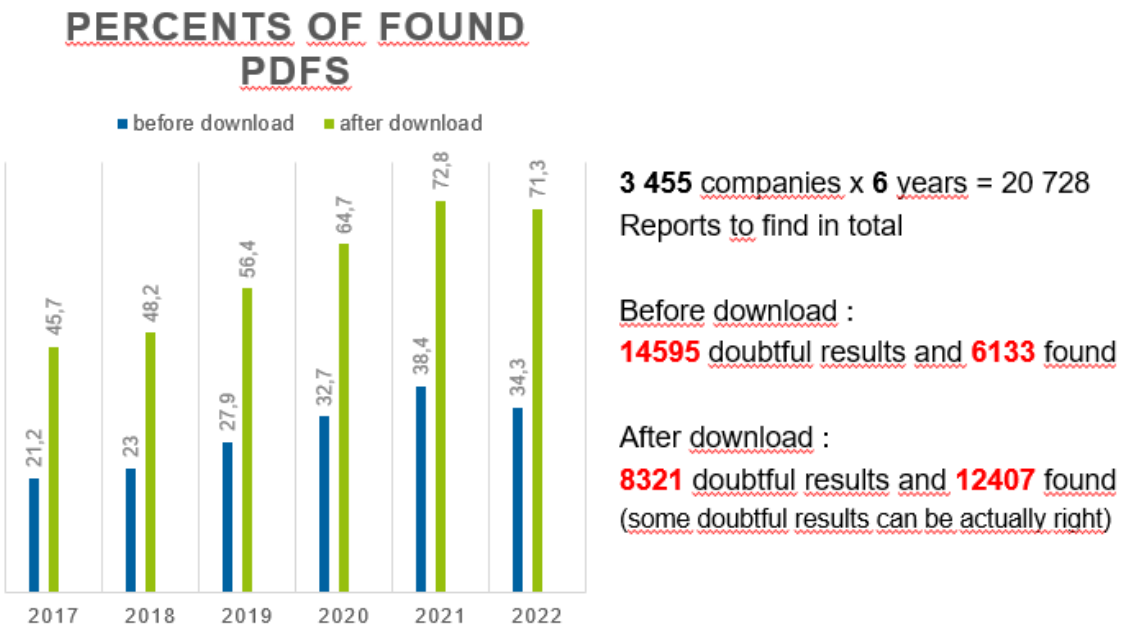

FIGURE 9 – Résultats du projet GAIA (nombre de PDFS trouvés et douteux)

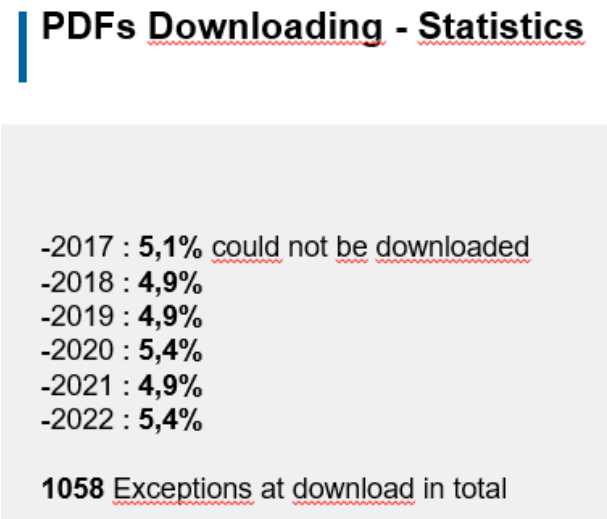

FIGURE 10 – Statistiques sur les PDFs n'ayant pu être téléchargés

Ces résultats étaient donc plutôt bons et ont satisfait l'équipe Gaia.

B.2 Projet ESCB Exchange

Pour ce projet, ma mission était très courte et devait être réalisée en 8 jours maximum, et en 5 jours elle était terminée.

Je devais donc reprendre le code écrit précédemment pour la comparaison bilatérale des sources de données, et mettre à jour la présentation existante avec les nouveaux résultats.

La fusion des 3 datasets sur le code ISIN des entreprises menait à un échantillion assez restreint d'observations, mais suffisant pour les représenter graphiquement.

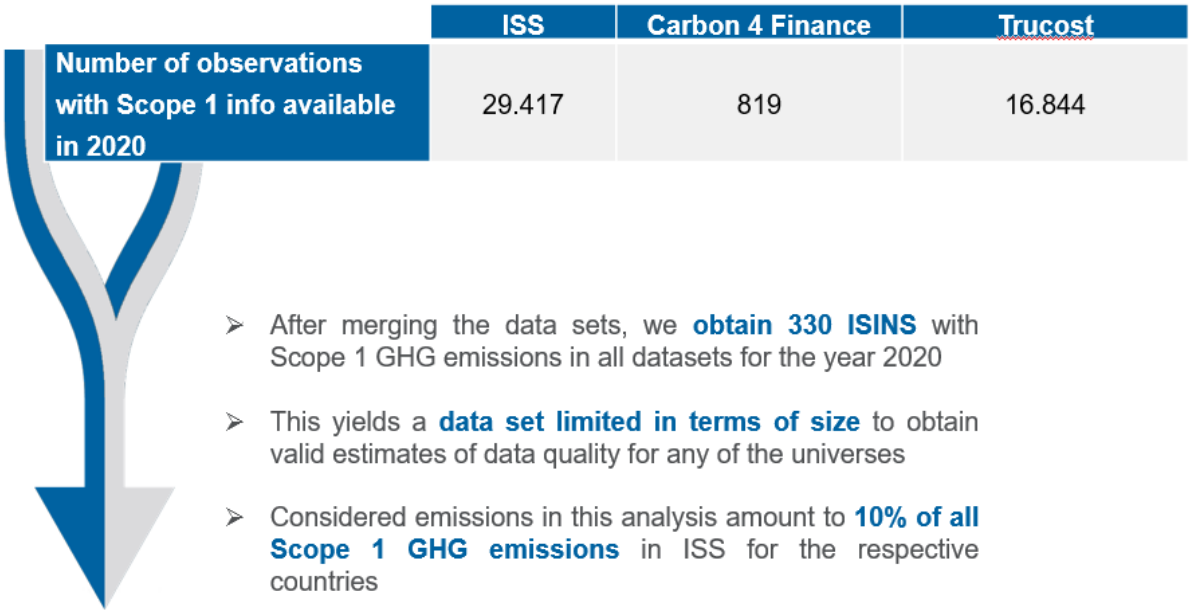

FIGURE 11 – Résultats de la fusion des datasets

Il fallait ensuite, pour chaque "Scope" (1, 2, 3) (= périmètres du bilan d'émissions de CO2) comparer les sources deux à deux graphiquement et obtenir le coefficient de correlation.

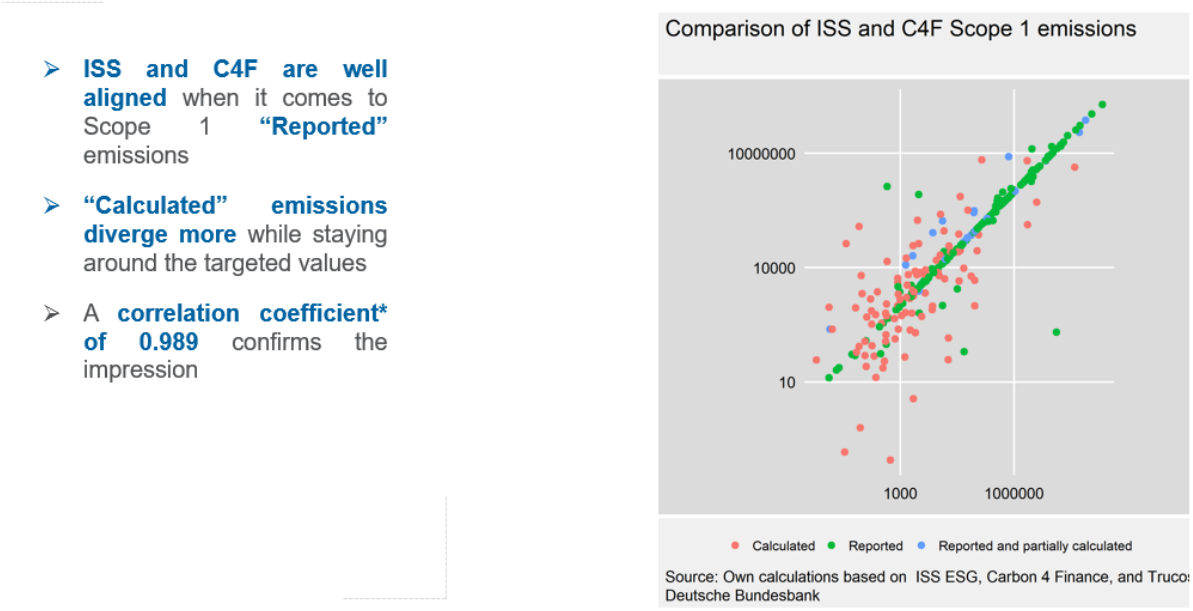

FIGURE 12 – Comparaison des sources ISS et Carbon4Finance

Afin de pouvoir déterminer le dataset contenant les données les plus justes, on cherche pour chaque scope le nombre d'observations pour lesquelles, dans une marge de tolérance de 10% :
- Les valeurs concordent toutes entre les trois sources de données
- Aucune source de données n'est d'accord avec les autres

Le reste des observations peuvent alors être comparées. Le datasource qui est le plus de fois en accord avec d'autres est le plus exact.

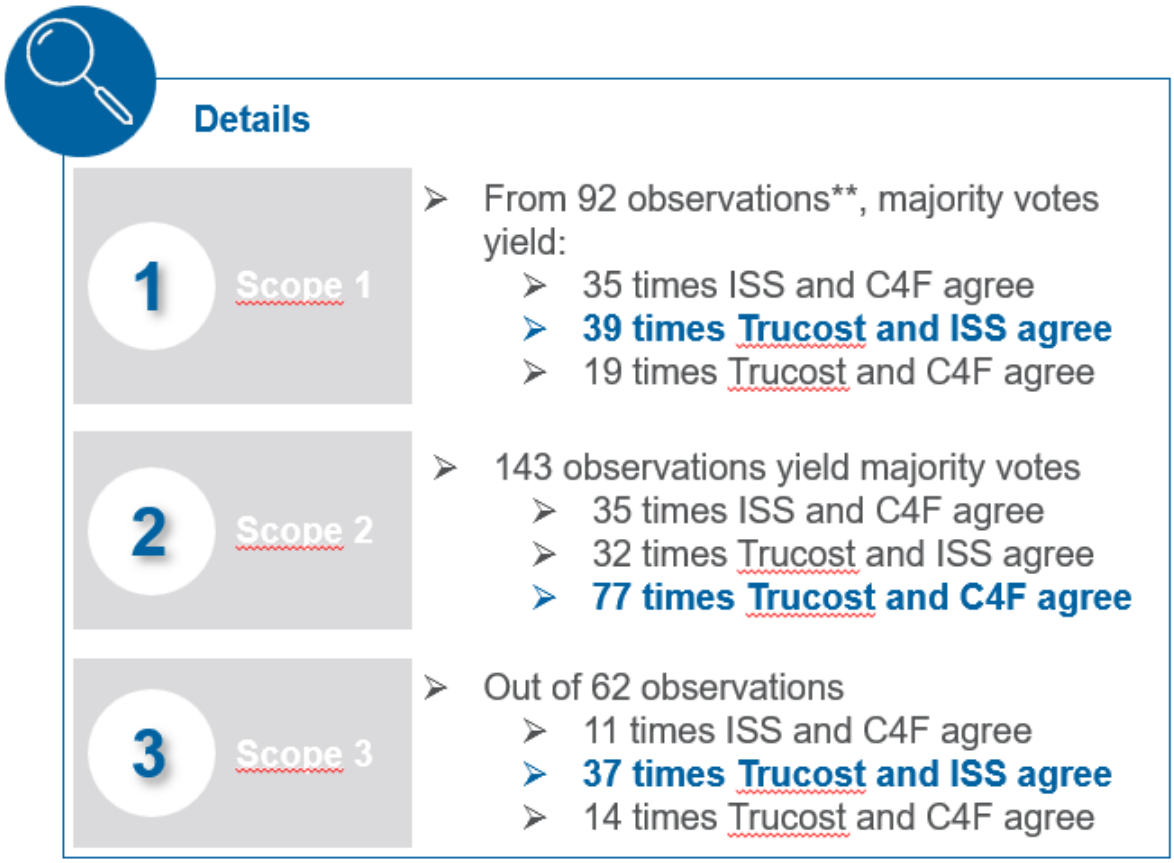

FIGURE 13 – Nombre d'observations identiques entre les sources

La principale difficulté durant ce projet était de comprendre le code d'une autre personne portant sur des opérations statistiques ainsi que d'où venaient cetains résultats notés dans la présentation.

Mon tuteur m'a ensuite suggéré d'essayer la comparaison quadrilatérale, avec en plus les données d'ERICA, mais seulement 10 observations restaient après la fusion. Les données plus récentes sont donc attendues.

B.3 Projet CSDB

Ma mission pour ce projet est donc de trouver une méthode de comparaison de datasets optimale et efficace.

J'ai dans un premier temps cherché les différentes méthodes qui existent en Python pour comparer facilement des datasets. Il y a notamment les méthodes "compare" et "equals" de pandas, mais celle-ci ne fonctionne que sur des datasets de forme identique. Cela n'était donc pas garanti, par exemple lorsque des colonnes étaient ajoutées sur une nouvelle version.

La package Datacompy en revanche, semblait remplir la mission à merveille, avec la possibilité de générer un rapport détaillé sur les différences et similitudes trouvées.

J'ai d'abord testé ces packages sur des petits Dataframes de test.

Le seul problème était que les lignes étaient vues comme différentes lorsqu'elles contiennent "" ou " ", de même lorsqu'un dataframe contient une colonne supplémentaire mais qui est vide (On veut qu'ils soient considérés dans ce cas comme identiques). Une fois ces détails résolus, j'ai pu tester la méthode de comparaison sur les premiers vrais fichiers csv, pesant plus de 2Go.

A la première exécution, je remarque que toutes les lignes sont différentes. Il fallait en fait trier les datasets par ISIN afin que la comparaison se fasse correctement. IL restait tout de même des fausses différences invisibles à l'oeil nu, à cause d'un nombre différent d'espaces (pouvant aller jusqu'à 150 espaces). Ceux-ci sont donc remplacés par None. Parfois des espaces à la fin des mots étaient aussi présents, il était donc nécessaire de les supprimer.

Concernant les nombres, ils pouvaient souvent arriver qu'ils soient arrondis seulement dans l'un des datasets. J'ai résolu cela en comparant les floats avec une tolérence de 1%. Pour 1 colonne, cette méthode n'a pas fonctionné, peut être à cause d'un format string. Ici, les mêmes entiers étaient succédés par un ".0" dans l'un des datasets. Après de longues recherches, je n'ai finalement pas trouvé de solution, celles trouvées cassant toutes les valeurs de toutes les colonnes avec un nombre inifini de "0".

Le dernier obstacle était les dates qui n'étaient pas du même format dans tous les datasets, étant parfois notés en toutes lettres. Les colonnes de dates ont finalement été toutes supprimées, n'étant pas importantes.

Au fil de l'utilisation du package Datacompy, j'ai découvert que l'on pouvait joindre les datasets sur une colonne spécifique, rendant le tri inutile.

Cependant, nous avons soupçonné que l'autre méthode de comparaison, un peu plus artisanale, serait plus optimale. En effet, il ne nécessite que très peu de lignes de code, mais chaque ligne est très longue à s'exécuter, calculant beaucoup d'informations non indispensables pour nous (20min en tout).

J'ai donc tenté de comparer les colonnes une à une avec les méthodes de pandas, et il s'est avéré que cette solution était beaucoup plus efficace (8min vs 20min d'exécution). Les colonnes supplémentaires étaient ensuite recherchées puis inspectées afin de voir si elles sont vides.

Tout fonctionnait parfaitement, à part ce problème de ".0" et pour certaines chaînes de caractères encodées différemment.

Pour terminer, j'ai exécuter mon code pour les 12 mois de 2015, et documenté les résultats dans un markdown. Pour ce faire, j'ai dû utiliser un Virtual Computer, car ces datasets faisaient entre 4 et 7Go.

```
### CSDB_200904_Datenbankabzug vs CSDB_200904_Datenbankabzug_v2

**csv reading time / filter / string strip :**
between 4min43 and 10min11

**sorting :** between 1m22 and 2min2s

**comparison of common columns :** 19 to 50s

### **Results :**

NAT_INS_CODE :
            self      otherc
121145  2544806.0    2544806
121146  2544817.0    2544817
121147  3522569.0    3522569
475136    4318884  4318884.0
475137    4318765  4318765.0
...           ...        ...
484764   560348.0     560348
484765   560359.0     560359
484766   560364.0     560364
484767   560368.0     560368
484768   560370.0     560370

[1998 rows x 2 columns]
SHORT_NAME :
                                      self                                  other
43871       RT OPTIMUM ÁNAK14 (A) ANT      RT OPTIMUM ÃNAK14 (A) ANT
43873       RT OPTIMUM ÁNAK14 (T) ANT      RT OPTIMUM ÃNAK14 (T) ANT
51482    HYPO PF LIQUIDITÂäT(A) ANT  HYPO PF LIQUIDITÃÂ¤T(A) ANT
51483    HYPO PF LIQUIDITÂäT(T) ANT  HYPO PF LIQUIDITÃÂ¤T(T) ANT
53291    GUT PENSIONSRÂÝCKST(T) ANT  GUT PENSIONSRÃÂCKST(T) ANT
...                              ...                           ...
3993779      Banco Popular EspaÂñol      Banco Popular EspaÃÂ±ol
3993826      Banco Popular EspaÂñol      Banco Popular EspaÃÂ±ol
3997600      Banco Popular EspaÂñol      Banco Popular EspaÃÂ±ol
```

FIGURE 14 – Extrait des résultats - Premiers datasets - avril 2009

Après cela, il m'a été conseillé de me concentrer sur le nouveau projet d'NLP, afin de ne pas avoir la tête sur plusieurs projets en même temps.

B.4 Projet NFIG

Tout comme le projet CSDB Exchange, celui-ci n'a duré que 5 jours pour la génération du rapport.

Avant cela, j'avais dû essayer de comprendre le code déjà écrit pour trouver automatiquement les fichiers générés dans le code, au moment où nous croyions encore avec mon collègue qua nous allions travailler là-dessus.

2 semaines plus tard, ma vraie mission a pu commencer :

A partir de 2 fichiers Excel comme ci-dessous, regroupant quel fichier output est généré à quelle ligne dans quel fichier .do et .log de quelle date, je devais donc générer un rapport avec Python selon le modèle de la figure 16.

date	file_to_be_sent	file_sent	freq	found_in_cmdline	concordance	to_be_found_in_dofile	line_number
20200918	summary_bista_corr_liab.tex	no	1			no code folder	
20200918	summary_bista_share_liab.tex	no	1			no code folder	
20200930	Figure_1_1_1_ts_aggregated_bymaturity_vjkre.png	no	2	Figure_1_1_1_ts_aggregated_bymaturity_vjkre	100	20200930_2_descriptive_analysis.do	32
20200930	Figure_1_2_1_ts_aggregated_allloan_vjkre.png	no	2	Figure_1_2_1_ts_aggregated_allloan_vjkre	100	20200930_2_descriptive_analysis.do	40
20200930	Figure_2_1_1_hist_share_inclmort.png	no	2	Figure_2_1_1_hist_share_inclmort	100	20200930_2_descriptive_analysis.do	63
20200930	Figure_2_2_1_hist_share_exludingmort.png	no	2	Figure_2_2_1_hist_share_exludingmort	100	20200930_2_descriptive_analysis.do	67
20200930	Figure_3_1_1_redeploy_index_epu_combined.png	no	2	Figure_3_1_1_redeploy_index_epu_combined	100	20200930_2_descriptive_analysis.do	235
20200930	Figure_3_1_1_redeploy_index_short_epu.png	no	2	Figure_3_1_1_redeploy_index_short_epu	100	20200930_2_descriptive_analysis.do	241
20200930	Figure_4_1_1_liabside_epu.png	no	2	Figure_4_1_1_liabside_epu	100	20200930_2_descriptive_analysis.do	397
20200930	Figure_5_1_1_liabside_exclovernight_epu.png	no	2	Figure_5_1_1_liabside_exclovernight_epu	100	20200930_2_descriptive_analysis.do	408
20200930	Figure_5_2_1_liabside_exclovernight_simple_epu.png	no	2	Figure_5_2_1_liabside_exclovernight_simple_epu	100	20200930_2_descriptive_analysis.do	412
20200930	Table_0_1_1_assetside_summary_allvars.tex	no	2	Table_0_1_1_assetside_summary_allvars	100	20200930_2_descriptive_analysis.do	150
20200930	Table_0_2_1_assetside_summary_loanlevel_bysector.tex	no	2	Table_0_2_1_assetside_summary_loanlevel_bysector	100	20200930_2_descriptive_analysis.do	173
20200930	Table_0_3_1_correlate_redeploy_index_epu.tex	no	2	Table_0_3_1_correlate_redeploy_index_epu	100	20200930_2_descriptive_analysis.do	252
20200930	Table_0_3_1_correlate_redeploy_index_epu.tex	no	2	Table_0_3_1_correlate_redeploy_index_epu	100	20200930_2_descriptive_analysis.do	428
20200930	Table_0_4_1_liabside_summary_allvars.tex	no	2	Table_0_4_1_liabside_summary_allvars	100	20200930_2_descriptive_analysis.do	329
20200930	Table_1_1_1_baseline_assetside_loanlevel_size_oct2020.tex	no	2	Table_1_1_1_baseline_assetside_loanlevel_size_oct2020	100	20200930_3_regressions.do	97
20200930	Table_2_1_1_baseline_assetside_loanshareinclmort_size_oct2020.tex	no	2	Table_2_1_1_baseline_assetside_loanshareinclmort_size	100	20200930_3_regressions.do	142
20200930	Table_3_1_1_baseline_assetside_loanshareexclmort_size_oct2020.tex	no	2	Table_3_1_1_baseline_assetside_loanshareexclmort_size	100	20200930_3_regressions.do	188
20200930	Table_4_1_1_baseline_liabside_liablevel_oct2020.tex	no	2	Table_4_1_1_baseline_liabside_liablevel_oct2020	100	20200930_3_regressions.do	326
20200930	Table_5_1_1_baseline_liabside_liabshare_oct2020.tex	no	2	Table_5_1_1_baseline_liabside_liabshare_oct2020	100	20200930_3_regressions.do	378
20200930	Table_5_2_1_baseline_liabside_liabshare_exclovernight_oct2020.tex	no	2	Table_5_2_1_baseline_liabside_liabshare_exclovernight_	100	20200930_3_regressions.do	429
20201006	Figure_1_1_1_ts_aggregated_bymaturity_vjkre.png	yes	2	Figure_1_1_1_ts_aggregated_bymaturity_vjkre	100	20201006_2_descriptive_analysis.do	32
20201006	Figure_1_2_1_ts_aggregated_allloan_vjkre.png	yes	2	Figure_1_2_1_ts_aggregated_allloan_vjkre	100	20201006_2_descriptive_analysis.do	40
20201006	Figure_2_1_1_hist_share_inclmort.png	yes	2	Figure_2_1_1_hist_share_inclmort	100	20201006_2_descriptive_analysis.do	63
20201006	Figure_2_2_1_hist_share_exludingmort.png	yes	2	Figure_2_2_1_hist_share_exludingmort	100	20201006_2_descriptive_analysis.do	67

FIGURE 15 – Fichier Excel à partir duquel le rapport devait être généré

Projekt-Id: 2020\0026
Datum: 20211025

log-Datei: 20211025_3_regressions_COVID_asset_loancatereg.log
do-Datei: 20211025_3_regressions_COVID_asset_loancatereg.do

Zeile do	log	n	Outputdatei
226	3562	1	pscore_match.png
395	4748	1	Table_6_1_1_covid_collapsed_did_corloan_save_credit_oct2021.tex
414	4826	1	Table_6_1_1_regression_summary_collapsed_treated_control_corloan_save_credit_oct2021.tex
513	5758	1	Table_6_1_1_covid_collapsed_did_corloan_save_credit_shortwindow_oct2021.tex
569	6155	2	all_combined_covid_corporate_loan_psmatched.png
738	7094	1	Table_6_3_1_covid_collapsed_did_corloan_save_credit_highexpo_political_oct2021.tex
830	7718	1	Table_6_3_1_covid_collapsed_did_corloan_save_credit_highexpo_political_shortwindow_oct2021.tex

log-Datei: 20211025_3_regressions_COVID_asset_VJKRE.log
do-Datei: 20211025_3_regressions_COVID_asset_VJKRE.do

Zeile do	log	n	Outputdatei
280	1061	1	Table_6_2_1_regression_summary_collapsed_treated_control_VJKRE_oct2021.tex
357	1741	2	Table_6_2_1_covid_collpased_did_VJKRE_save_credit_2secexpo_oct2021.tex
449	2397	1	Table_6_2_1_covid_collpased_did_VJKRE_save_credit_2secexpo_shortwindow_oct2021.tex

FIGURE 16 – Exemple de rapport type à générer automatiquement

Comme le modèle de rapport le montre, la 1ère colonne du tableau est elle-même composée de 2 colonnes, ce qui impliquait d'avoir des dataframes emboîtés. Cela a été un peu compliqué à mettre en place, mais a finalement bien fonctionné à la fin, en choisissant un format HTML plutôt que Markdown pour le rapport.

Egalement, la colonne "n" fait référence à la colonne "freq" dans l'Excel, mais il n'était pas clair si ce nombre n'était valable que pour 1 apparition (autrement dit, 3 apparitions dans 1 fichier = 3 fréquences), par exemple si le nom du fichier output apparait plusieurs fois à la même ligne ; ou si une fréquence était valable pour le nombre total des apparitions, comme le montre le modèle.

Dans le doute, j'ai choisi la 1ère option, plus compliquée à implémenter, et le résultat ressemble à cela :

Datum: 20201006
log-Datei: 20201006_2_descriptive_analysis.log
do-Datei: 20201006_2_descriptive_analysis.do

Zeile		n	Outputdatei
log	do	[]	[]
[49.0, 66.0]	[32.0]	[2, 2, 2]	Figure_1_1_1_ts_aggregated_bymaturity_vjkre.png
[62.0, 66.0]	[40.0]	[2, 2, 2]	Figure_1_2_1_ts_aggregated_allloan_vjkre.png
[453.0, 497.0]	[63.0]	[2, 2, 2]	Figure_2_1_1_hist_share_inclmort.png
[453.0, 505.0]	[67.0]	[2, 2, 2]	Figure_2_2_1_hist_share_exludingmort.png
[2480.0]	[235.0]	[2, 2]	Figure_3_1_1_redeploy_index_epu_combined.png
[2489.0]	[241.0]	[2, 2]	Figure_3_1_1_redeploy_index_short_epu.png
[5078.0]	[397.0]	[2, 2]	Figure_4_1_1_liabside_epu.png
[5102.0]	[408.0]	[2, 2]	Figure_5_1_1_liabside_exclovernight_epu.png
[5110.0]	[412.0]	[2, 2]	Figure_5_2_1_liabside_exclovernight_simple_epu.png
[666.0]	[150.0]	[2, 2]	Table_0_1_1_assetside_summary_allvars.tex
[1844.0]	[173.0]	[2, 2]	Table_0_2_1_assetside_summary_loanlevel_bysector.tex
[2515.0, 5144.0]	[252.0, 428.0]	[2, 2, 2, 2]	Table_0_3_1_correlate_redeploy_index_epu.tex
[2640.0]	[329.0]	[2, 2]	Table_0_4_1_liabside_summary_allvars.tex

FIGURE 17 – Extrait du rapport généré automatiquement

J'ai demandé au client ce qu'il en était pour ce doute, il devait lui-même vérifier mais ne m'a jamais redit.

B.5 Projet de recherche NLP

Pour commencer tout travail de recherche, il faut débuter par la revue littéraire en anglais pour trouver des articles similaires. J'en ai donc trouvé 8 qui se rapporchaient de près ou de loin à notre sujet, soit recherchant juste des datasets mentionnés dans les papiers de recherche, soit les méthodologies, ou utilisant un LLM. Ce que nous souhaiterions dans l'idéal atteindre est de trouver seulement les datasets et méthodologies utilisées, et non simplement mentionnées.

La 2nde tâche était de diviser les papiers de recherche en phrases, afin de pouvoir plus tard les fournir une à une à ChatGPT. Pour cela, j'ai utilisé le package de NLP nltk, mais ai dû ajouter à la main quelques règles, notamment pour "et al." et "N." pour numéro, où les points étaient là considérés comme des fins de phrases. J'ai alors dû mettre en place une fonction récursive, ce qui était assez délicat mais très formateur. Installer le package NLTK n'était également pas des plus simple à la Bundesbank, certaines dépendances devaient être téléchargées à la main, dû aux diverses autorisations.

Une fois cela terminé, ma mission était de libeller quelques phrases contenant des mentions de datasets et d'autres n'en contenant pas, puis de fournir ces phrases à ChatGPT et d'observer les résultats. Concernant la question exacte à poser à ChatGPT, il était plus

efficace de lui donner la définition d'un dataset ou d'une méthodologie, puis de lui demander de s'appuyer sur cette définition pour trouver les éléments. Cependant, toute suite de nombres était considérée comme un dataset, or cela ne nous intéressait pas, souhaitant seulement les noms des datasets utilisés, et non ceux fabriqués durant la recherche.

Un autre département de la Bundesbank (IT7) était intéressé par le projet et lors de la réunion avec eux, nous avons pu présenter ces résultats. Pour des raisons budgétaires, il était finalement préférable selon eux de diviser les papiers en pages plutôt qu'en phrases afin de pas devoir fait trop de petites requêtes.

J'ai donc de nouveau sélectionné des pages contenant des datasets ou méthodologies, et d'autres ne contenant rien, documenté le tout dans un Excel, et envoyé le fichier à l'IT7 afin qu'ils piussent tester les échantillons facilement eux-même.

L'intention de cette recherche devait être officiellement déposée à un comité de recherche, il fallait donc soumettre l'Abstract assez tôt. Une collègue l'a écrit, et le reste de l'équipe devait le relire et corriger, ce à quoi j'ai participé.

Cette même collègue a aussi commencé l'écriture d'un code visant à filtrer les papiers pour supprimer les parties non pertinentes, notamment les pages Titre, les Références, etc... Comme j'étais censée mettre en commun nos 2 codes pour préparer des extraits directement utilisables pour ChatGPT mais qu'il n'était pas terminé, j'ai décidé de le faire moi-même. Elle avait, dans un Jupyter Notebook, préparer les différentes sections pour le code à venir, par exemple "Suppression de l'Introduction", "Division en sections", etc...

A ce moment s'est présentée une phase de réflexion sur ce que nous souhaitions vraiment, et s'il était vraiment pertinent de supprimer autant de parties. En me renseignant auprès de Sebastian, mon référant technique pour ce projet, j'ai appris que les datasets se référant à un sondage conduit pour la recherche comptent, ainsi que les parties de datasets utilisées (Ex : La 1ère vague de tel dataset). Plus spécifiquement, j'ai appris que l'Introduction ne devait pas être supprimée, ni les descriptions de Tableaux ou Figures.

Dans un premier temps, j'ai tenté de trouver un moyen de supprimer les tableaux de fichiers PDF. Certains packages Python existaient pour extraire les tableaux en Dataframes, mais ne permettaient pas de les supprimer du fichier, et surtout ne reconnaissaient pas les tableaux contenant seulement des bordures de lignes et non de colonnes. D'autres packages ayant l'air intéressant ne pouvaient simplement pas être installés à la Bundesbank, recquérant trop de dépendances ou autres logiciels (ex : Anaconda).

Après de longues recherches, j'ai conclu qu'il était nécessaire d'obtenir un format structuré à partir des PDF.

J'ai tenté de trouver un convertisseur Markdown, mais aucun n'existait pour Python, sauf payant. J'ai ensuite essayé de convertir en HTML, mais les balises résultantes étaient toutes des , y compris pour les tableaux.

Finalement, ma solution était de convertir les PDF en DOCX (package pdf2docx). Les tableaux y étaient après directement détectables, et grâce au package docx, il était possible de faire toutes sortes de modifications sur les docx, y compris supprimer les tableaux. De nombreuses possibilités s'ouvraient alors : il était désormais possible de trouver les

caractères en gras (toujours des titres) et de supprimer le texte en 2 titres. J'ai donc pu supprimer la section Références beaucoup plus efficacement et en moins de lignes que ce qu'avait précédemment fait ma collègue. Le lien présent en bas de chaque a également pu être ainsi supprimé. A la fin, le docx pouvait être reconvertir en PDF grâce au package docx2pdf. La conversion en docx n'a simplement pas fonctionné sur 1 pdf sur 6. J'ai donc aussi traité les exceptions.

Pour ce qui est de la suppression de la page titre, j'ai simplement repris le code de ma collègue qui fonctionnait sur les papiers en format txt découpés en pages grâce à Fitz, car le format docx ne reconnait pas les numéros de pages. Il y a donc 2 phases de filtrage.

Sebastian a en parallèle écrit un code visant à communiquer avec l'API de ChatGPT qui lui envoie les extraits de textes et enregistre les réponses dans des txt, ce qui est très utile pour automatiser les requêtes et avoir accès à plus d'options. La prochaine étape était donc l'incorporation de mon filtre au processus de questionnement de ChatGPT. J'ai pu dans ce contexte apprendre à utiliser les classes en Python. Une difficulté que j'ai rencontrée était de comprendre le code déja fait et de s'y adapter le mieux possible pour incorporer proprement mon code.

C Planning général suivi

Conclusion

J'ai beaucoup appris sur le traitement des exceptions et l'écriture / lecture d'un fichier / utiliser l'API Dropbox, mais surtout faire un Google Crawler

Durant ce stage chez Gaea21, j'aurai donc réussi avec mon équipe à terminer les principales fonctionnalités de la version 0 du site Répertoire Vert.

Ainsi, une entreprise peut à présent :

- S'inscrire et se connecter
- Visualiser son profil et modifier ses informations
- Désactiver son compte
- Ajouter des produits et/ou services
- Visualiser ses produits/services, les supprimer et les modifier
- Visualiser ses statistiques, comme le nombre de clics sur ses produits, sans bug
- Parcourir les différentes catégories et sous-catégories, et voir la liste des entreprises appartenant à chaque sous-catégorie.
- Visualiser les profils des autres entreprises ainsi que leurs produits/services

Mes compétences se sont nettement améliorées en Git, que je sais maintenant utiliser en ligne de commande. Je sais désormais mettre un site en production, et coder avec Symfony et ReactJS.

J'ai pu apprendre à gérer une équipe et un projet avec des deadlines serrées, en utilisant des outils de gestion.

Mes collègues étaient pour la plupart plus âgés que moi, mais j'étais la plus ancienne sur le projet (à cause d'un turn-over important). De ce fait, je connaissais assez le projet pour pouvoir les accompagner. Mais la différence d'âge ne se ressentait pas, travailler avec eux était très fluide. Il y avait toujours un climat d'entraide et beaucoup de communication dans l'équipe.

Les réunions journalières avec mon tuteur et l'équipe me permettait de suivre leur avancée, et additionnellement si besoin, nous nous appelions avec un ou plusieurs membres, par exemple s'ils avaient besoin d'aide ou que quelque chose devait être rectifié. Nous communiquions beaucoup par Skype. Aussi, je mettais en commun notre travail plusieurs fois par semaine, ce qui me permettait de bien suivre la progression du projet.

L'outil de gestion que j'utilisais pour planifier était un tableau Google Sheet présentant les tâches, leur difficulté, et leur durée estimée que je remplissais avec mon équipe.

Avoir une certaine responsabilité était très formateur et m'a donné une idée du métier de Lead Programmer, qui me plairait d'ailleurs beaucoup. Cependant, je souhaiterais m'orienter dans le domaine de l'IA et du Big Data, c'est pourquoi je choisirai la spécialité Valorisation des Connaissances pendant mon cursus.

Bibliographie

[1] Vladimir Agafonkin. *Leaflet - a JavaScript library for interactive maps*. [en ligne]. Mis à jour le 4 septembre 2020 [Consulté le 14 décembre 2021]. URL : https://leafletjs.com/.

[2] Massimiliano Arione. « KnpLabs/KnpPaginatorBundle : SEO friendly Symfony paginator to sort and paginate ». In : *Github*. [en ligne]. Mis à jour le 2 décembre 2021 [Consulté le 17 décembre 2021]. URL : https://github.com/KnpLabs/KnpPaginatorBundle.

[3] Jeff Atwood et Joël Spolsky. *Stack Overflow - Where Developers Learn, Share, and Build Careers*. [en ligne]. Mis à jour le 30 décembre 2021 [Consulté le 17 décembre 2021]. URL : https://stackoverflow.com/.

[4] Ghaida Bouchâala. « Transform your excel data into a relational database : Symfony | Medium ». In : *Ghaida Bouchâala – Medium*. [en ligne]. Mis à jour le 5 juillet 2020 [Consulté le 22 novembre 2021]. URL : https://ghaidabouchala.medium.com/import-excel-data-in-the-database-symfony-back-end-e14efea51cd2.

[5] Doctrine. *Welcome to Doctrine 2 ORM's documentation ! - Doctrine Object Relational Mapper (ORM)*. [en ligne]. Mis à jour le 21 décembre 2021 [Consulté le 17 décembre 2021]. URL : https://www.doctrine-project.org/projects/doctrine-orm/en/2.10/index.html.

[6] Facebook. *React – Une bibliothèque JavaScript pour créer des interfaces utilisateurs*. [en ligne]. Mis à jour le 22 mars 2021 [Consulté le 8 novembre 2021]. URL : https://fr.reactjs.org/.

[7] Flickity. *Flickity · Touch, responsive, flickable carousels*. [en ligne]. Mis à jour le 19 décembre 2021 [Consulté le 2 décembre 2021]. URL : https://flickity.metafizzy.co/.

[8] Grafikart. *Tutoriels et Formations vidéos sur le développement web*. [en ligne]. Mis à jour le 28 décembre 2021 [Consulté le 18 août 2021]. Disponible sur : URL : https://grafikart.fr/.

[9] Theo Lampert. « react-flickity-component - npm ». In : *npm*. [en ligne]. Mis à jour le 30 août 2021 [Consulté le 8 novembre 2021]. URL : https://www.npmjs.com/package/react-flickity-component.

[10] Symfony SAS. *Symfony, High Performance PHP Framework for Web Development*. [en ligne]. Mis à jour le 29 décembre 2021 [Consulté le 17 décembre 2021]. URL : https://symfony.com/doc/current/index.html.

[11] SymfonyCasts. *SymfonyCasts - PHP and Symfony Video Tutorial Screencasts*. [en ligne]. Mis à jour le 23 décembre 2021 [Consulté le 23 août 2021]. URL : https://symfonycasts.com/.

[12] Alexander Weissman. *Getting Started | Select2 - The jQuery replacement for select boxes*. [en ligne]. Mis à jour le 3 mai 2020 [Consulté le 28 septembre 2021]. URL : https://select2.org/.

Table des figures