Hackathon SmartDoc.ai

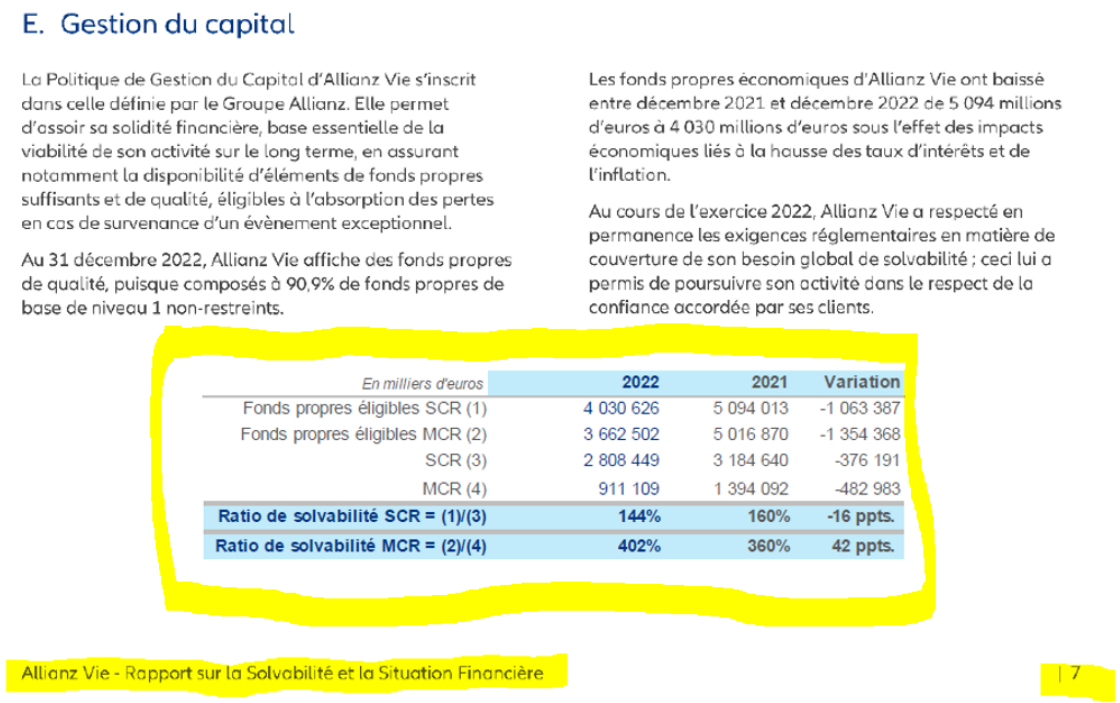
Un récapitulatif des liens utiles est dispo vers la fin de ce doc

Présentation de la problématique

Dans les sujets NLP, un Data Scientist est fréquemment confronté à la lecture de documents dans divers formats, notamment au format PDF. Les défis rencontrés concernent souvent les résultats produits par les OCR [1] utilisés pour lire le contenu des documents, en particulier :

1. La présence d'informations inutiles, telles que **les bas de pages et les hauts de pages,** dans le texte **reconstruit** à partir de l'OCR.

2. La rétention d'informations non structurées provenant de tableaux, nécessitant souvent une séparation du contenu contextuel des paragraphes du contenu des tableaux et des graphes.



E. Gestion du capital

La Politique de Gestion du Capital d'Allianz Vie s'inscrit dans celle définie par le Groupe Allianz. Elle permet d'assoir sa solidité financière, base essentielle de la viabilité de son activité sur le long terme, en assurant notamment la disponibilité d'éléments de fonds propres suffisants et de qualité, éligibles à l'absorption des pertes en cas de survenance d'un évènement exceptionnel.

Au 31 décembre 2022, Allianz Vie affiche des fonds propres de qualité, puisque composés à 90,9% de fonds propres de base de niveau 1 non-restreints.

Les fonds propres économiques d'Allianz Vie ont baissé entre décembre 2021 et décembre 2022 de 5 094 millions d'euros à 4 030 millions d'euros sous l'effet des impacts économiques liés à la hausse des taux d'intérêts et de l'inflation.

Au cours de l'exercice 2022, Allianz Vie a respecté en permanence les exigences réglementaires en matière de couverture de son besoin global de solvabilité ; ceci lui a permis de poursuivre son activité dans le respect de la confiance accordée par ses clients.

En milliers d'euros	2022	2021	Variation
Fonds propres éligibles SCR (1)	4 030 626	5 094 013	-1 063 387
Fonds propres éligibles MCR (2)	3 662 502	5 016 870	-1 354 368
SCR (3)	2 808 449	3 184 640	-376 191
MCR (4)	911 109	1 394 092	-482 983
Ratio de solvabilité SCR = (1)/(3)	**144%**	**160%**	**-16 ppts.**
Ratio de solvabilité MCR = (2)/(4)	**402%**	**360%**	**42 ppts.**

Allianz Vie - Rapport sur la Solvabilité et la Situation Financière | 7

Les objectifs de cet exercice sont :

1. **Récupérer uniquement le contenu utile des rapports, excluant les informations indésirables notamment les bas de pages, les hauts de pages, les informations provenant des tableaux et des graphes. (Un bonus sur la partie 1 est détaillé dans la suite)**
2. **Construire une architecture RAG capable de répondre à la trame de questions fournies**

[1] OCR : logiciels de Reconnaissance Optique de Caractères

Les rapports que nous traitons pour cet exercice, sont les rapports SFCR des assureurs. Ce sont des rapports publiés périodiquement par les compagnies d'assurance, pour fournir des informations sur la situation financière, la solvabilité et la gestion des risques de l'entreprise.

Nous avons préalablement utilisé pour vous, l'OCR API Vision de Google Cloud Platform (GCP) pour extraire le contenu brut des rapports SFCR au format JSON. Vous trouverez les rapports au format PDF dans le sous dossier **data/pdfs/**. Vous trouverez également les fichiers JSON produits par l'OCR dans le sous dossier **data/ocr/**

L'avantage de l'utilisation de l'API Vision de GCP, réside dans la possibilité d'obtenir des informations supplémentaires telles que la taille des caractères et la position des caractères.

Déroulement de l'exercice :

Première partie :

Nous vous fournissons le code **« helper.py »**, qui contient la fonction « ***produce_brut()*** » pour exploiter les fichiers JSON générés par l'OCR. Cette fonction produit un tableau Excel des blocs textuels inclut dans le rapport, avec un maximum d'informations sur le rapport. Les informations inclues dans le rapport sont définies dans le bloc suivant de la fonction :

```
f = {}
#f["para_id"] = para_id
f["num_page"]=num_page
f["text"] = text
f["width"] = max(x_list) - min(x_list)
f["height"] = max(y_list) - min(y_list)
f["area"] = f["width"] * f["height"]
f["chars"] = len(text)
f["char_size"] = f["area"] / f["chars"] if f["chars"] > 0 else 0
f["pos_x"] = (f["width"] / 2.0) + min(x_list)
f["pos_y"] = (f["height"] / 2.0) + min(y_list)
f["aspect"] = f["width"] / f["height"] if f["height"] > 0 else 0
f["layout"] = "h" if f["aspect"] > 1 else "v"
f["x0"]=x_list[0]
f["x1"]=x_list[1]
f["y0"]=y_list[0]
f["y1"]=y_list[1]
page_features.append(f)
```

Vous avez la liberté de modifier le bloc (dans l'image ci-dessus), si nécessaire, afin de peaufiner les statistiques existantes ou d'incorporer d'autres données pertinentes à vos yeux pour la suite de l'exercice.

Après application de la fonction ***produce_brut()*** à un JSON produit par l'OCR, vous aurez un exemple de tableau suivant :

num_page	text	width	height	area	chars	char_size	pos_x	pos_y	aspect	layout	x0	x1	y0	y1	assureur
1	Allianz	0,25042	0,03088	0,00773	9	0,00086	0,18067	0,11283	8,10976	h	0,05546	0,30588	0,09739	0,09857	allianz
1	ALLIANZ VIE	0,11765	0,01069	0,00126	12	0,0001	0,11429	0,21556	11,0066	h	0,05546	0,17311	0,21021	0,21021	allianz
1	Rapport sur la solvabilité	0,66387	0,04157	0,0276	27	0,00102	0,39244	0,27138	15,9707	h	0,0605	0,72437	0,25653	0,25059	allianz
1	Exercice 2022	0,12101	0,01069	0,00129	14	9,2E-05	0,11597	0,43765	11,321	h	0,05546	0,17647	0,4323	0,4323	allianz
1	et la situation financière	0,63866	0,03325	0,02124	27	0,00079	0,61345	0,32185	19,2053	h	0,29412	0,93277	0,30641	0,30523	allianz
3	Conformément aux Articles 51 et 25	0,82689	0,07601	0,06285	661	9,5E-05	0,48403	0,10689	10,8788	h	0,07059	0,89748	0,06888	0,06888	allianz
3	Le présent Rapport sur la Solvabilité	0,79664	0,02257	0,01798	228	7,9E-05	0,46891	0,16805	35,3037	h	0,07059	0,86723	0,15677	0,15677	allianz
3	Les informations présentées dans le	0,71597	0,06295	0,04507	361	0,00012	0,42689	0,2215	11,3744	h	0,06891	0,78487	0,19002	0,19002	allianz
3	Table des matières	0,1916	0,01188	0,00228	19	0,00012	0,16639	0,31948	16,1325	h	0,07059	0,26218	0,31354	0,31354	allianz
3	Synthèse	0,07731	0,01306	0,00101	9	0,00011	0,10924	0,35451	5,91781	h	0,07059	0,1479	0,34798	0,34917	allianz
3	A. Activité et résultats	0,20168	0,0095	0,00192	25	7,7E-05	0,16975	0,38599	21,2269	h	0,06891	0,27059	0,38124	0,38124	allianz
3	A.1 . Activité	0,09748	0,00831	0,00081	16	5,1E-05	0,13277	0,41508	11,7253	h	0,08403	0,18151	0,41093	0,41093	allianz
3	A.2 .	0,02353	0,00831	0,0002	6	3,3E-05	0,0958	0,43527	2,83025	h	0,08403	0,10756	0,4323	0,43112	allianz
3	A.3 .	0,02521	0,00713	0,00018	6	3E-05	0,09664	0,45606	3,53782	h	0,08403	0,10924	0,45249	0,45249	allianz
3	A.4 .	0,02521	0,00831	0,00021	6	3,5E-05	0,09664	0,47684	3,03242	h	0,08403	0,10924	0,47268	0,47268	allianz
3	A.5 . Autres informations	0,18151	0,0095	0,00172	27	6,4E-05	0,17479	0,49762	19,1042	h	0,08403	0,26555	0,49287	0,49287	allianz

Votre tâche consiste à mettre en place une fonction capable de :

- Détecter le contenu inutile des rapports, notamment les bas de pages, les hauts de pages et le contenu des tableaux.
- Identifier les paragraphes.
- Repérer les grands titres des différentes parties du rapport.

La fonction que vous développerez prendra en entrée l'output de `produce_brut()` et ajoutera une nouvelle colonne, "Label", au tableau résultant. Les valeurs de cette colonne seront soit "Inutile", "Paragraphe", ou "Titre".

Il est impératif que cette fonction puisse être généralisée pour une labélisation automatique applicable à d'autres rapports SCFR. Nous mettons à votre disposition plusieurs exemples de SFCR ainsi que leurs rapports correspondants pour faciliter vos tests et permettre une solution robuste et généralisable.

Bonus première partie : Extraction lisible des informations des tableaux

Dans cette partie optionnelle, nous vous invitons à réfléchir et, si possible, à implémenter une méthode permettant d'extraire les informations contenues dans les tableaux des rapports SFCR de manière lisible et structurée.

- Objectifs :
    - Identification des tableaux : Détecter automatiquement les sections des rapports correspondant à des tableaux.
    - Extraction structurée : Extraire les données des tableaux en conservant leur sens et leur structure (par exemple, sous forme de dataframe ou tableau Excel).
    - Présentation lisible : Proposer une manière claire de restituer ces informations, comme un tableau lisible ou un résumé.
- Contraintes : Les tableaux peuvent contenir des informations complexes, comme des titres de colonnes imbriqués ou des valeurs fusionnées. Tenez compte de ces spécificités pour garantir une extraction cohérente. Vous pouvez utiliser les informations supplémentaires fournies par l'API Vision de Google Cloud Platform (taille et position des caractères) pour aider à détecter et structurer les tableaux.
- Évaluation : Les solutions proposées seront évaluées sur :
    - Leur capacité à détecter et structurer les tableaux avec précision.
    - La lisibilité des données extraites.

    - L'originalité et la robustesse de la méthode.
- Indications : Vous pouvez vous inspirer de modèles d'extraction existants ou explorer des approches basées sur les positions des blocs textuels et la segmentation visuelle. Si l'implémentation complète est trop complexe, décrivez clairement votre méthode proposée, avec des exemples et des pistes d'amélioration.

Deuxième partie :

Dans cette partie, vous devrez développer une architecture RAG (Retrieval-Augmented Generation) permettant de répondre à un ensemble de questions prédéfinies à partir des rapports SFCR. Vous êtes libre de personnaliser les paramètres en fonction des besoins spécifiques. Voici les étapes principales attendues :

- Choix du modèle LLM : Expliquez pourquoi vous avez choisi un modèle LLM spécifique (par exemple, GPT-3, Mistral, ou un autre modèle open-source ou closed-source).
- Stratégie de chunking : Définissez une méthode pour découper les rapports en sections ou morceaux (chunks).
- Modèle d'embedding : Précisez le modèle utilisé pour la vectorisation des chunks (par exemple, Sentence-BERT, OpenAI embeddings, etc.).
- Paramètres supplémentaires : Décrivez tout autre paramètre ou choix technique important (indexation, moteur de recherche vectoriel, etc.).
- Trame de questions à prendre en considération : Utilisez le fichier **Trame_questions.pdf** pour évaluer votre RAG, nous allons réajuster la trame de questions pour l'évaluation de vos architectures RAG.

Justifiez ces choix pour garantir des performances optimales.

Livrables attendus :

- Code source : Votre implémentation complète (chaque partie sur un notebook jupyter avec le code helper.py réajusté si besoin), y compris les explications/commentaires pour chaque étape.
- Rapport (Readme sur Github) ou présentation PPT : Présentez votre architecture et justifiez vos choix techniques. Illustrez si possible vos résultats avec des exemples.
- Dépôt GitHub : Forkez notre dépôt initial : https://github.com/AlumniECC/Hackathon_Smartdoc.ai et ajoutez vos travaux. Partagez le lien vers votre dépôt sur l'excel suivant : https://centralecasablanca-my.sharepoint.com/:x:/g/personal/imad_zaoug_centrale-casablanca_ma/EWvYqsFs2oBKoSWg2X0Q2zcBStATPMiXvYKxVztwwfC3mA?rtime=kJgIE4gU3Ug .

Indications sur les deux parties :

- Vous pouvez explorer des stratégies basées sur la taille des caractères et la longueur des blocs pour proposer une première solution de détection des titres, des paragraphes et des contenus inutiles.
- Vous pouvez utiliser la première partie comme stratégie de chunking (chunking par sections)
- Toute initiative ou proposition pertinente sera grandement appréciée et pourra faire la différence dans l'évaluation de votre test.
- La propreté du code ainsi que sa documentation seront également des critères évalués.
- Nous vous souhaitons bon courage et restons à votre disposition pour toute clarification nécessaire.

Récapitulatif des liens :

Lien de l'excel pour renseigner vos liens repo github :	https://centralecasablanca-my.sharepoint.com/:x:/g/personal/imad_zaoug_centrale-casablanca_ma/EWvYqsFs2oBKoSWg2X0Q2zcBStATPMiXvYKxVztwwfC3mA?rtime=kJgIE4gU3Ug
Lien de notre repo Github :	https://github.com/AlumniECC/Hackathon_Smartdoc.ai