Projet Gestion de Produit Structuré

BERNA Jules, BLENEAU Quentin,
KADRI Nassim & LAMURE Thibaut

3A IF

Mars 2024

Sommaire

1 Introduction

Ce projet a pour objectif de construire une application qui aide le gérant d'un produit structuré à effectuer son travail quotidien.

L'un des intérêts du projet était rassembler les deux parcours de la filière IF autour d'un projet qui conclut l'année. Ce projet rassemble beaucoup de choses vu et apprises par les deux côtés et semble se rapprocher fortement d'une situation professionnelle envisageable.

2 Comment lancer le projet

Une fois l'archive dézippée, vous pouvez installer l'environnement Streamlit à l'aide de la commande suivante :

```
pip install streamlit
```

Ensuite, pour lancer l'application web, il vous suffit de vous placer à la racine du projet et d'exécuter la commande suivante :

```
python -m streamlit run src/python/main.py
```

A la suite de cela, la page web de l'application devrait s'ouvrir sur votre navigateur par défaut.

3 Partie pricing en C++

Concernant le pricing en C++ du projet, nous nous sommes concentrés sur la réutilisation du code d'un précédent projet afin de faciliter le début. Le code utilisé correspond au projet que les MEQA ont eu en marché de taux. Nous avons donc notamment utilisé la librairie PNL de Jérôme Lelong.

Pour ce qui est de l'architecture, elle n'a pas été beaucoup modifiée. La classe Pricer correspond au cœur de cette partie. Elle possède entre autres un attribut Position et un attribut MonteCarlo. C'est avec cette classe que les informations nécessaires sont renvoyées côté Python dans un JSON. La classe Position possède en attribut les prix, deltas et écart-types (volatilités) associés. Ces valeurs sont calculées dans la classe MonteCarlo avec différentes méthodes. Dans cette classe, on retrouve notamment un attribut de type YosemiteOption (3ème produit du sujet qui nous était attribué) et un attribut GlobalModel. La classe YosemiteOption possède une méthode payoff qui renvoie pour n'importe quel instant le

payoff de l'option.

La classe GlobalModel permet de simuler différents Assets et différents taux de change grâce au modèle de Black & Scholes. La classe DataFeeder permet de lire et de stocker le JSON avec les informations sur le produit et le marché, envoyé par Python. Cette classe, qui est également un reste du précédent projet, a été la partie la plus difficile à adapter. Il était cependant nécessaire de la garder car cela nous permettait de ne pas toucher aux autres classes hormis YosemiteOption.

Les principales difficultés rencontrées concernent le fait d'adapter de manière intelligente ce que nous avions fait dans les projets précédants, notamment bien adapter la classe DataFeeder. De plus, la couverture et la gestion des flux n'étaient plus la même que précédemment. Cependant, il a été plutôt facile de réutiliser une majorité du précédent projet.

4 Partie applicative en Python

En ce qui concerne la partie front-end, notre choix s'est attardé sur la plateforme open-source Streamlit qui permet de créer rapidement des interfaces web permettant de présenter des résultats. Cette plateforme possède de nombreux outils pour la mise en forme des dataframes Pandas ce qui nous a amplement simplifié le travail. En effet, nous avons énormément utilisé la librairie Pandas en Python qui permet de manipuler facilement des grands jeu de données.

Pour revenir à Pandas, cette librairie possède des fonctions générant directement des dataframes à partir de fichiers Excel ou CSV. Cela a été très pratique car les données de marché ainsi que les données de pricing/hedging données par le pricer en C++ étaient sous ce format.

Il n'a pas été très compliqué d'apprendre à utiliser Streamlit car c'est très intuitif, et l'un des membres de l'équipe s'en était déjà servi au cours d'un stage. Pour l'élève du groupe en I2MF, cela a été nouveau, car nous n'avons pas réutilisé les technologies web apprises au cours de cette année, notamment React, car le cadre du projet ne le nécessitait pas forcément. Nous avons jugé qu'il était plus efficace d'utiliser une technologie comme Streamlit.

Nous pensons que cela est très instructif car nous pensons maintenant être en mesure de fournir assez rapidement une application web intuitive dans le cadre d'un stage en front-office par exemple.

5 Conclusion

Ce projet a permis de faire communiquer les différentes compétences acquises au cours de cette dernière année. De plus, il nous a permis de comprendre comment pricer/hedger un nouveau type de produit : un produit structuré. Nous sommes désormais également en mesure de fournir des applications efficientes dans le domaine financier, et ce grâce à notre double compétence finance quantitative et ingénierie informatique. Nous sommes heureux d'avoir réalisé ce projet car celui nous a permis de nous rendre compte encore plus du genre de travail que nous pouvons être amené à faire dans notre futur travail. En réalité le travail en entreprise reprend les mêmes étapes que nous avons pu mettre en place dans ce projet, à savoir une demande d'un client, une partie de réflexion sur le produit en question et comment utiliser les bonnes technologies pour résoudre le problème, mettre en place la partie d'implémentation en équipe, et enfin à travers notre vidéo explicative, présenter notre produit au client et montrer à quel point celui-ci répond à sa demande initiale. C'est la première que nous sommes amenés à mettre en place toutes ces étapes dans le domaines financiers. De plus le fait de pouvoir s'appuyer sur des anciens projets n'est pas anodin. En effet, dans l'industrie, il est rare de reconstruire des outils de zéro, et il est souvent utile d'utiliser des anciennes techniques et outils, les améliorer, pour pouvoir répondre le plus efficacement à la problématique. De même, la découverte de nouvelles technologies comme nous avons pu les rencontrer en tant qu'élèves en MEQA, telles que des outils pour le front end.