**Documentation : Outil de Reporting Python**

Notre outil de reporting de gestion de portefeuille est composé des 5 fichiers suivants.

1. Service.py
2. Data.py
3. Model.py
4. View.py
5. Tool\_Management.py

L’objectif de cette documentation est de vous expliquer comment fonctionne notre outil afin de rendre son utilisation le plus clair possible.

1. **Fichier Service :**

Ce fichier contient un dictionnaire nommé « config » qui permet de stocker dans le même fichier .py toutes les données en « dur » utilisées par notre outil. En effet, dans ce dictionnaire vous trouverez :

* Les chemins d’accès des fichiers Excel utilisés.
  + Il faut les mettre à jour avec vos chemins d’accès aux fichiers
  + Ne pas oublier de transformer les « \ » en « / »
* Les paramètres utilisés pour réalisés notre simulation de monte carlo
  + num\_simulations
  + risk\_free\_rate (dans notre cas, nous avons utilisé un taux sans risque de 0, car au Q4 de 2019, le taux du bund allemand 10 ans était très proche de 0)
* Les paramètres utilisés pour réaliser les calculs de VaR et CVaR.
  + confidence\_level
  + num\_days

L’intérêt d’un tel fichier est d’externaliser les données en dur de notre code afin de faciliter leur modification.

1. **Fichier Data :**

Ce fichier python contient la classe « DataManagement » qui permet de récupérer et retraiter si nécessaire les données de marché des fichiers Excel. Elle comprend des méthodes pour lire et traiter des données provenant de diverses sources, filtrer et transformer les données à des fins de rapport et d'analyse.

Initialisation :

Constructeur (\_\_init\_\_) :

* + Initialise les chemins des fichiers pour les données de l'indice CAC40, les données du marché boursier, les prix de clôture et les données de reporting en utilisant un dictionnaire de configuration (config).

Méthodes privées :

* \_\_set\_companies\_features(df)
  + Extrait les noms de sociétés, les ISINs et les tickers uniques d'un DataFrame.
  + Retourne un DataFrame contenant ces caractéristiques.
* \_\_set\_analysis\_period(start\_date)
  + Calcule la période d'analyse, commençant trois ans avant la date de début donnée jusqu'à la date de début elle-même.
  + Retourne les dates de début et de fin de la période d'analyse.
* \_\_get\_columns\_todrop(df)
  + Identifie les colonnes avec plus de 25% de valeurs manquantes dans un DataFrame.
  + Retourne une liste de colonnes à supprimer.
* \_\_convert\_excel\_date(excel\_date)
  + Convertit un format de date Excel en un objet datetime standard.
  + Retourne la date convertie.
* \_\_convert\_dates(date\_str)
  + Tente de convertir une chaîne de date (ou une date Excel) en un format de date standard.
  + Retourne la chaîne de date formatée.

Méthodes publiques

* get\_reporting\_data(features=False)
  + Lit les données de reporting à partir d'un fichier Excel.
  + Transforme les données pour avoir une ligne par date pour chaque entreprise et supprime les colonnes et les lignes avec des valeurs manquantes excessives.
  + Si features est vrai, retourne également un DataFrame avec les caractéristiques des entreprises.
* get\_CAC40\_stock\_market(features=False)
  + Lit les données du marché boursier CAC40 à partir d'un fichier CSV.
  + Transforme les données de manière similaire à get\_reporting\_data.
  + Si features est vrai, retourne également un DataFrame avec les caractéristiques des entreprises.
* get\_CAC40\_index\_historical\_price(start\_date, end\_date)
  + Lit les prix historiques de l'indice CAC40 à partir d'un fichier CSV.
  + Filtre les données pour la plage de dates donnée et les trie par date.
  + Retourne un DataFrame avec les prix de clôture ajustés.
* get\_CAC40\_stock\_historical\_price(features=False)
  + Lit les prix historiques des actions CAC40 à partir d'un fichier Excel.
  + Convertit les dates et ajuste les colonnes pour les ISINs.
  + Si features est vrai, retourne un DataFrame avec les caractéristiques des entreprises. Sinon, retourne les données des prix des actions.
* get\_portfolio\_reporting\_data(isin\_list, start\_date, end\_date)
  + Filtre les prix historiques des actions CAC40 pour la liste d'ISIN donnée et la plage de dates.
  + Supprime les lignes avec des valeurs manquantes.
  + Retourne le DataFrame filtré.
* get\_portfolio\_analysis\_data(isin\_list, start\_date)
  + Filtre les prix historiques des actions CAC40 pour la liste d'ISIN donnée dans la période d'analyse.
  + Supprime les lignes avec des valeurs manquantes.
  + Retourne le DataFrame filtré.

1. **Fichier Model :**

La classe PortfolioManagement étend la classe DataManagement pour fournir des fonctionnalités de gestion de portefeuille avancées. Elle inclut des méthodes pour effectuer des simulations de Monte Carlo, optimiser les portefeuilles en fonction du ratio de Sharpe, calculer les rendements cumulatifs, et évaluer les risques financiers à l'aide de la Value at Risk (VaR) et de la Conditional Value at Risk (CVaR).

**Méthodes privées :**

* **\_\_daily\_return(df)**
  + Calcule les rendements quotidiens d'un DataFrame de données.
  + Retourne un DataFrame des rendements quotidiens.
* **\_\_monte\_carlo\_simulation(daily\_returns)**
  + Effectue une simulation de Monte Carlo pour générer plusieurs portefeuilles aléatoires.
  + Retourne des listes de poids, rendements moyens, volatilités et ratios de Sharpe pour chaque simulation.
* **\_\_get\_monte\_carlo\_simulation\_results(daily\_returns)**
  + Utilise les résultats de la simulation de Monte Carlo pour créer un DataFrame de résultats.
  + Retourne un DataFrame avec les poids des actifs, le rendement, la volatilité et le ratio de Sharpe pour chaque portefeuille simulé.
* **\_\_get\_max\_sharpe\_ratio\_portfolio(daily\_returns)**
  + Identifie le portefeuille avec le ratio de Sharpe maximal parmi les résultats de la simulation de Monte Carlo.
  + Retourne une série représentant le portefeuille optimal.
* **\_\_get\_best\_portfolio(isin\_list, start\_date)**
  + Obtient le portefeuille optimal en fonction des rendements historiques et du ratio de Sharpe.
  + Retourne une série représentant le portefeuille optimal.
* **\_\_get\_optimal\_weights(isin\_list, start\_date)**
  + Obtient les poids optimaux pour le portefeuille en fonction du portefeuille optimal.
  + Retourne une série de poids optimaux.
* **\_\_compute\_portfolio\_daily\_return(isin\_list, start\_date, end\_date)**
  + Calcule le rendement quotidien du portefeuille sur une période donnée en fonction des poids optimaux.
  + Retourne un DataFrame avec les valeurs liquidatives et les rendements quotidiens du portefeuille, ainsi que les poids optimaux.
* **\_\_get\_cumulative\_return(isin\_list, start\_date, end\_date)**
  + Calcule les rendements cumulatifs du portefeuille sur une période donnée en fonction des rendements quotidiens.
  + Retourne un DataFrame avec les rendements cumulatifs du portefeuille.
* **\_\_compute\_Value\_at\_Risk(daily\_returns, weights)**
  + Calcule la Value at Risk (VaR) et la Conditional Value at Risk (CVaR) pour le portefeuille en fonction des rendements quotidiens et des poids optimaux.
  + Retourne la VaR et la CVaR.

**Méthodes publiques :**

* **get\_consolidated\_quarter\_reporting(isin\_list, start\_date, end\_date)**
  + Génère un rapport consolidé trimestriel pour le portefeuille basé sur les performances, la volatilité, la VaR et la CVaR.
  + Exporte les résultats au format Excel.
  + Retourne un DataFrame avec les statistiques consolidées du trimestre.
* **reporting(isin\_list, start\_date, end\_date)**
  + Génère un rapport détaillé comprenant les valeurs liquidatives, les poids optimaux, et les rendements cumulatifs du portefeuille.
  + Retourne un tuple contenant le DataFrame des valeurs liquidatives, les poids optimaux et les rendements cumulatifs.

1. **Fichier View :**

Cette application de reporting financier a été développée en utilisant Python et les bibliothèques suivantes :

* tkinter pour l'interface graphique
* tkcalendar pour la sélection des dates
* pandas pour la manipulation des données
* matplotlib pour la visualisation des données
* pptx pour générer des présentations PowerPoint
* win32com.client pour convertir des fichiers PowerPoint en PDF

**Classe : FinancialReportingApp**

**Constructeur \_\_init\_\_(self)**

* Initialise l'application avec un titre, des dimensions spécifiques et un fond.
* Initialise un gestionnaire de données (DataManagement) et les variables pour les entreprises sélectionnées et les dates de début et de fin.
* Crée les widgets de l'interface utilisateur.

**Méthode create\_widgets(self)**

* Crée la structure principale de l'interface utilisateur, y compris le cadre de titre et les onglets pour différentes fonctionnalités :
  + Détermination du portefeuille
  + Overview et composition du portefeuille
  + Visualisation graphique du portefeuille et de son benchmark

**Méthode create\_determination\_tab(self, notebook)**

* Crée l'onglet de détermination du portefeuille avec :
  + Un cadre pour la sélection des dates
  + Une liste des entreprises disponibles
  + Une liste des entreprises sélectionnées pour le portefeuille
  + Des boutons pour valider le portefeuille, générer les données et créer des fichiers PowerPoint et PDF

**Méthode create\_overview\_tab(self, notebook)**

* Crée l'onglet d'overview avec :
  + Deux tables pour afficher les données consolidées et la composition du portefeuille

**Méthode display\_portfolio\_overview(self)**

* Affiche les données consolidées du portefeuille et la composition du portefeuille dans les tables de l'onglet overview.

**Méthode create\_visualization\_tab(self, notebook)**

* Crée l'onglet de visualisation avec deux espaces pour afficher les graphiques de l'évolution de la valeur liquidative du portefeuille et du benchmark CAC40.

**Méthode plot\_portfolio\_graph(self, ax)**

* Trace l'évolution de la valeur liquidative du portefeuille sur un graphique.

**Méthode plot\_benchmark\_graph(self, ax)**

* Trace l'évolution historique du prix ajusté du CAC40 sur un graphique.

**Méthode create\_date\_selectors(self, parent)**

* Crée les sélecteurs de date pour choisir les dates de début et de fin de l'analyse.

**Méthode create\_company\_list(self, parent)**

* Crée la liste des entreprises disponibles pour le portefeuille, permettant la sélection multiple et l'ajout d'entreprises au portefeuille.

**Méthode create\_selected\_portfolio(self, parent)**

* Crée la liste des entreprises sélectionnées pour le portefeuille avec la possibilité de supprimer des entreprises.

**Méthodes add\_selected\_companies(self) et remove\_selected\_companies(self)**

* Ajoutent ou suppriment des entreprises de la liste des entreprises sélectionnées pour le portefeuille.

**Méthode update\_selected\_companies(self)**

* Met à jour l'affichage de la liste des entreprises sélectionnées.

**Méthode update\_list\_isin(self)**

* Met à jour la liste des ISINs des entreprises sélectionnées et les dates de début et de fin.

**Méthode generate\_list\_isin(self)**

* Génère la liste des ISINs des entreprises sélectionnées en fonction de la liste des entreprises et des dates choisies.

**Méthode generate\_portfolio\_data(self)**

* Génère et affiche les données du portefeuille et les visualisations correspondantes.

**Méthode display\_portfolio\_visualization(self)**

* Affiche les graphiques de l'évolution de la valeur liquidative du portefeuille et du benchmark CAC40 dans l'onglet de visualisation.

**Méthode save\_charts\_as\_images(self)**

* Sauvegarde les graphiques du portefeuille et du benchmark en tant qu'image.

**Méthode generate\_powerpoint(self)**

* Génère une présentation PowerPoint incluant :
  + Une diapositive de titre
  + Une diapositive avec l'overview et la composition du portefeuille
  + Une diapositive avec les visualisations graphiques du portefeuille et du benchmark

**Méthode generate\_pdf(self)**

* Convertit la présentation PowerPoint générée en un fichier PDF.

1. **Fichier Tool\_Management :**

Ce fichier permet de lancer notre outil python depuis le module « FinancialReportingApp ». De ce fait, pour utiliser notre outil, il suffit d’ouvrir le fichier « Tool\_Management », et de lancer le code. En exécutant ce code, la « mainloop » associée au module « FinancialReportingApp » s’exécutera, et l’affichage Tkinter commencera.

Veuillez vous rendre le fichier word « Mode Opératoire » pour apprendre à utiliser notre outil.