Documentation outil PRIIPS

Le code est composé de 5 parties :

* PRIIPS
* scenario\_performance\_builder (constructeur des objets scénarios de performances)
* scenario\_performance (Classe de base qui permet d’hérité les attributs et fonctions de base)
* scenario\_performance\_1 (Classe qui contient les calculs relatif au catégories 1)
* scenario\_performance\_2 (Classe qui contient les calculs relatif au catégories 2)
* scenario\_performance\_3 (Classe qui contient les calculs relatif au catégories 3)
* scenario\_performance\_4 (Classe qui contient les calculs relatif au catégories 4)

# scenario\_performance\_builder

Le builder est l’objet qui permet d’instancier un objet d’une classe précise. Les deux objets instanciables sont les scénarios de calcules des performances pour les catégories 1,2,3 et 4.

Chaque catégorie utilise des arguments différents, mais qui se présente de la même manière.

Les deux arguments qui doivent être contenu doit être :

* dict\_matrix\_yield : Le dictionnaire qui associe pour chaque type de scénario la matrice de rendement des taux brutes. Les clés doivent être « favorable » ou « défavorable ». Le programme s’attend que les matrices soient de la forme (simulation x Projection)
* dict\_of\_parametre : Le dictionnaire qui contient les différents paramétrages du run. Certains éléments de ce dictionnaire sont obligatoires.
  + « rhp » : période historique recommander (>0)
  + « category » : type de de catégorie (1,2,3 ou 4)
  + « mnt\_investment\_initial » : Montant initialement investit par le client (Règlementairement et arbitrairement le montant est de 10 000€)
  + Autres arguments qui dépendent du type de scénario de performances.
* is\_logging\_activate : Boolean qui indique si le logging est activé. Ce paramètre est optionnel, et par défaut égal à False. Il permet de créer un document txt qui contient les logs. (Actuellement uniquement les erreurs et des warnings sont intégrés mais pas les résultats intermédiaires).

L’instanciation se fait de la manière suivante :

La première étape consiste en la création de l’objet associé au builder.

obj\_builder\_scn\_perf = ScenarioPerformanceManager()

La seconde étape consiste à crée l’objet voulu en fonction des paramètres mis en entrée.

scn\_perf = obj\_builder\_scn\_perf.create\_object(

    matrix\_yield=matrix\_yield,

    dict\_of\_parametre=dict\_parametre\_scenario\_performance

)

Une fois réalisée, l’objet scn\_perf contient un objet de la classe Scenario\_performance\_« category » en fonction du paramétrage de « category » dans le dictionnaire de paramètre, (ie Scenario\_performance\_4 pour la categorie 4).

Lors de l’instanciation des objets des tests sont réalisés. La classe parente contient une abstract méthode appelé « get\_category\_mapping\_dictionary\_for\_funct\_performance ». Cette méthode permet d’associé la fonction de run de chaque objet de chaque classe à la méthode exécutable ainsi qu’au méthode de test. Afin que l’association se réalise correctement il est nécessaire que les méthodes de test contiennent le terme « test » dans la clé associé au dictionnaire de mapping de la fonction. En effet toutes les clés contenants le mot « test » seront appelées pour réaliser des vérifications.

La fonction de l’exécution des scénarios de performance (brute) doit avoir pour clé : « performance\_funct ».

scn\_perf.performance\_funct()

Cette ligne renvoie pour chaque classe les scénarios de performances brutes

scn\_perf.performance\_funct\_net()

Cette ligne renvoie pour chaque classe les scénarios de performances nets. Cette méthode est développée pour la catégorie 4 seulement.

# Scenario\_performance\_4

Le calcul des scénarios intègre le calcul des coûts. C’est pourquoi les rendements en entrés doivent être des frais bruts et que des informations soit précisés sur le contrat et les frais que l’on y applique.

## Input

En plus des arguments minimaux qui sont :

* « rhp » : période historique recommander,
* « category » : type de de catégorie (4),
* « mnt\_investment\_initial » : Montant initialement investit par le client (Règlementairement et arbitrairement le montant est de 10 000€),

Il faut rajouter les arguments suivants :

Pour les rendements (« matrix\_yield ») :

* Des rendements où l’on a enlevé la PB discrétionnaire utilisé dans le cas favorable.
* Des rendements où l’on conserve la politique de PB utilisé dans le cas défavorable et intermédiaire.

Pour la gestion de informations du contrat, par défaut la valeur est de 0 :

* « TMG » : taux minimum garantie ou le « TMGA » pour taux minimum garantie annuel. Ces taux doivent être net.
* « type\_of\_support » : Le type de support du fond euro, soit il est sur un Monosupport soit sur un Multisupport. Par defaut le type de support est « Monosupport »

Pour la gestion des frais, les informations sont enregistrées au sein d’un dictionnaire (dict\_taxes), par défaut la valeur est de 0 :

* « rate\_chargement\_euro » : taux de frais des chargements sur encours du fond euro
* « rate\_admission\_fees » : taux de frais pour admissibilité du fond euro (frais d’entré)
* « rate\_management\_fees\_on\_outstandings » : taux de frais de la gestion des contrats
* « rate\_fees\_on\_global\_actif » : taux de frais sur les actifs gérés
* « cost\_of\_transaction » : taux liées aux transactions
* « other\_cost » : autre coûts
* « cost\_fees\_linked\_to\_results » : coût liée aux résultats de performance
* « cost\_linked\_to\_incentive\_commissions » : coût liée aux commissions d’incitation

Il est possible d’ajouter les coûts aux fils du temps qui permet d’intégrer les coûts tels que les frais de transaction et de performances. La variable est « est\_cout\_fil\_du\_temps\_integre » par défaut la valeur est False.

Pour les quantiles associés aux performances, les informations sont enregistrées au sein d’un dictionnaire, dict\_quantil :

* Les clés doivent contenir les mots fav, defav ou middle encerclé de « \_ » ou de rien. Les exemples suivants fonctionnent : threshold\_trust\_for\_fav\_perf ou fav mais ne fonctionne pas pour thresholdtrustforfavperf.
* Si par défaut aucune valeur n’est fixée dictionnaire vide ou manquant alors un dictionnaire est généré contenant les quantiles selon la réglementation de 2021.
* Les valeurs des quantiles doivent être des pourcentages.

Les informations relatives aux quantiles sont présentes au sein de la fonction « get\_dict\_parametre\_quantil\_perf ».

Les fonctionnalités de la classe scenario\_performance\_4 sont les suivantes :

* Bootstrap d’une matrix mise en entré
* Calcul des scénarios de performances

## Bootstrap

La fonction Bootstrap renvoie un matrice, np.ndarrray, de dimension (nombre de simulation, dimension de la matrice en enlevant la dimension des simulation)

La fonction prend en argument la matrice à Bootstrap, l’indice de la colonne à Bootstrap ainsi que le nombre de simulation à Bootstrap.

Par exemple, pour une matrice de taille (nombre de maturité=M, nombre de ténors=T, Nombre de scénario aléatoire=N) si on veut 100 simulations issues du Bootstrat alors la commande est la suivante :

scn\_perf.bootstrap(

    matrix\_to\_bootstrat=matrix\_yield,

    indice\_of\_shape\_to\_bootstrap=2,

    number\_of\_bootstrap=100

    )

Rappel, les listes en python commencent à 0 et non à 1, donc l’indice de taille de la matrice à Bootstrap doit lui aussi être soustrait de 1 par rapport à sa place dans la matrice.

La méthode réalise un Bootstrap et s’adapte à la colonne choisit pour réaliser le dit Bootstrap. Pour rappel, le Bootstrap est une méthode de tirage avec remises des simulations.

## Performance\_funct

La méthode permet de réaliser le calcul des scénarios économiques et renvoie un tuple contenant les 4 scénarios de la façon suivantes : fav\_perf, defav\_perf, middle\_perf, tension\_perf.

### Scénario favorable

En utilisant la matrice des rendements dont la PB discrétionnaire est enlevée, le calcul suivant est appliqué :

Pour un date t qui indique l’année de la projection sélectionner et un seuil α.

Où

* est la fonction quartil de coefficient α.(99%)
* est le rendement en t des simulations selon la situation favorable

### Scénario intermédiaire

En utilisant la matrice des rendements dont la PB discrétionnaire est enlevée, le calcul suivant est appliqué :

Pour un date t qui indique l’année de la projection sélectionner et un seuil α.

Où

* est la fonction quartil de coefficient α.(50%)
* est le rendement en t des simulations selon la situation intermediaire

### Scénario défavorable

En utilisant la matrice des rendements dont la PB discrétionnaire est enlevée, le calcul suivant est appliqué :

Pour un date t qui indique l’année de la projection sélectionner et un seuil α.

Où

* est la fonction quartil de coefficient α.(95%)
* est le rendement en t des simulations selon la situation favorable

### Scénario tension

Le scénario sous tension n’utilise pas les scénarios mais seulement les informations du contrat.

# Scenario\_performance\_1 et 3

Les classes scenaio\_performance\_1 et scenaio\_performance\_3, calculs les scénarios de performance pour les actifs de catégorie 1, respectivement 3. Les calculs intègrent les calculs nets. Les rendements historiques sont bruts. Ces méthodes héritent des formules de la classe scenario\_performance\_4.

A l’exception du calcul de la performance sous tension qui utilise les répartitions des rendements au quantile 1% pour une RHP de 1 et 5% sinon.

Les autres calcules sont identiques.

# Scenario\_performance\_2

La classe scenaio\_performance\_2, calculs les scénarios de performance pour les actifs de catégorie 2. Les calculs n’intègrent pas le calcul net. Les rendements historiques sont bruts. Il faut dans second temps enlevé les coûts.

## Input

En plus des arguments minimaux que sont :

* « rhp » : période historique recommander,
* « category » : type de de catégorie (2 ou 4),
* « mnt\_investment\_initial » : Montant initialement investit par le client (Règlementairement et arbitrairement le montant est de 10 000€),
* « data\_frequency » : fréquence des données historiques en inputs. Cette variable peut prendre que 3 valeurs : "daily prices", "weekly prices", "monthly prices". **Il n’y pas annuel !** Cette information est fournie par la réglementation.
* « matrix\_yield » : les rendements peuvent être soit log soit brut, le calcul n’est pas réalisé au sein de la classe. La réglementation demande que les rendements soient log.

Il faut rajouter les arguments suivants :

Pour les quantiles associés aux performances, les informations sont enregistrées au sein d’un dictionnaire, dict\_quantil :

* Les clés doivent contenir les mots fav et defav encerclé de « \_ » ou de rien. Les exemples suivants fonctionnent : threshold\_trust\_for\_fav\_perf ou fav mais ne fonctionne pas pour thresholdtrustforfavperf.
* Si par défaut aucune valeur n’est fixée dictionnaire vide ou manquant alors un dictionnaire est généré contenant les quantiles selon la réglementation de 2021.

Les informations relatives aux quantiles sont présentes au sein de la fonction « get\_dict\_parametre\_quantil\_perf ».

## Performance\_funct

La méthode permet de réaliser le calcul des scénarios économiques et renvoie un tuple contenant les 4 scénarios de la façon suivantes : fav\_perf, defav\_perf, middle\_perf, tension\_perf.

### Scénario favorable

Où

* Ou .
* : le nombre de périodes de négociation durant la période de calcul (si t = RHP alors n= N).
* : le centile de la loi Normale à , avec (changement réalisé en 2021)
* : la volatilité mesurée à partir de la distribution des rendements.
* : le coefficient d’asymétrie mesuré à partir de la distribution des rendements.
* : le coefficient d’excès d’aplatissement mesuré à partir de la distribution des rendements.
* : la moyenne des rendements

### Scénario intermédiaire

Où

* Ou .
* : le nombre de périodes de négociation durant la période de calcul (si t = RHP alors n= N).
* : la volatilité mesurée à partir de la distribution des rendements.
* : le coefficient d’asymétrie mesuré à partir de la distribution des rendements.
* : la moyenne des rendements

### Scénario défavorable

Où

* Ou .
* : le nombre de périodes de négociation durant la période de calcul (si t = RHP alors n= N).
* : le centile de la loi Normale à , avec
* : la volatilité mesurée à partir de la distribution des rendements.
* : le coefficient d’asymétrie mesuré à partir de la distribution des rendements.
* : le coefficient d’excès d’aplatissement mesuré à partir de la distribution des rendements.
* : la moyenne des rendements

### Scénario tension

Une image contenant texte, capture d’écran, Police, document

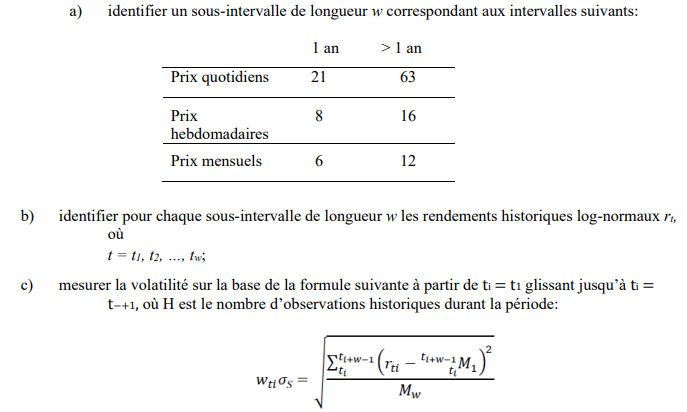
Description générée automatiquement

P.18 Annexe VI Règlement délégué 2021/2268 :

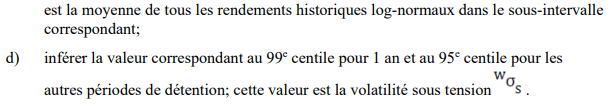
https://eur-lex.europa.eu/legal-content/FR/TXT/PDF/?uri=CELEX:32021R2268&from=FR

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement



La table du point a) est enregistré au sein de la méthode : « get\_dict\_parametre\_interval\_to\_calcul\_vol\_tension ». Cette méthode ne peut pas être modifié sans changer le code !



P.20-21 Annexe VI Règlement délégué 2021/2268 :

https://eur-lex.europa.eu/legal-content/FR/TXT/PDF/?uri=CELEX:32021R2268&from=FR

* : la volatilité sous tension calculée à partir de l’historique du support
* et pour les périodes de détention de 1 an.
* et pour les périodes de détention supérieures à 1 an