Sara Boutigny, 14/11/2020

Partie 1

Le code est structuré en une partie 1 correspondant au codage des stratégies de trading et la partie 2 d'optimisation de portefeuille.

Le calcul des signaux de chaque stratégie se fait grâce à la fonction strategy_signal qui prend en paramètres un dataframe de prix, un n, et le type de stratégie (momentum/contrariante, long only/long short, bollinger). La fonction renvoie un dataframe avec des 0 et des 1 en cas de stratégie long only et des 1 et -1 en cas de stratégie long short. Ces dataframe sont tous rangés dans un dictionnaire nommé strat_collection.

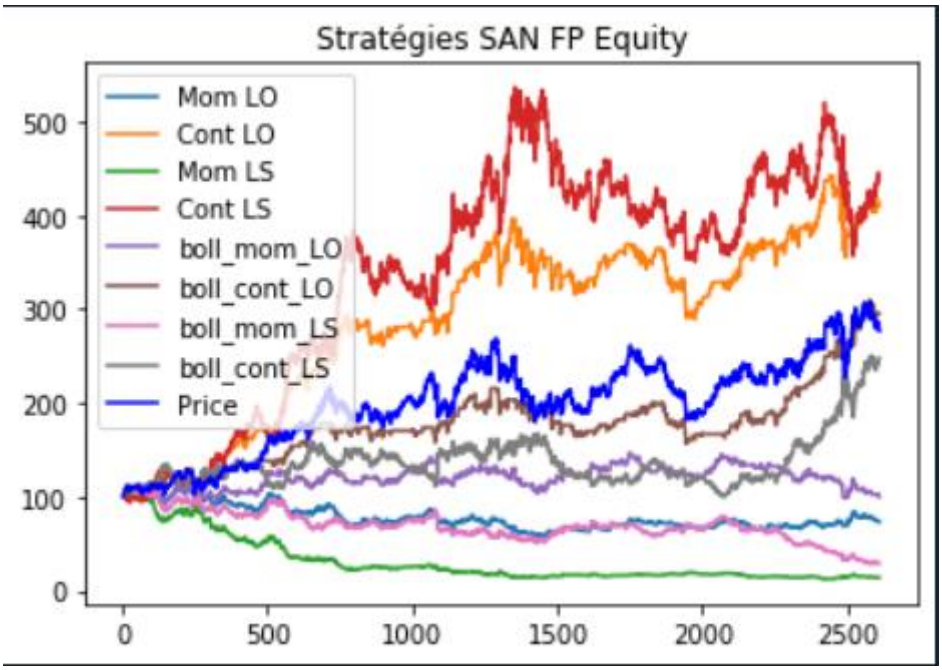
Par la suite, tous les calculs sont faits dans des dictionnaires qui contiennent des dataframe correspondant aux différentes stratégies. Ainsi, il suffit de boucler sur les items et les clés du dictionnaire de stratégies « strat_collection » qui contient les signaux d'achat/vente et d'effectuer les différents calculs : performances, performances rebasées, ratio rendement/volatilité. Et ce, pour chaque n. A la fin de ce processus, nous récupérons dans un dictionnaire n_opt les différents dataframe qui représentent, pour chaque stratégie, le n qui maximise le ratio rendement/volatilité.

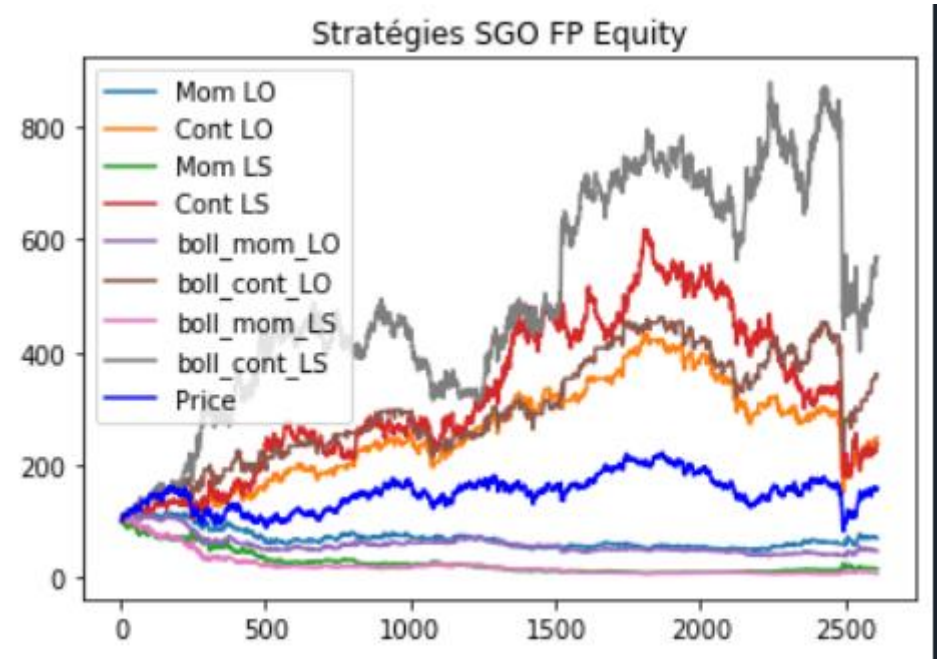
Par exemple, pour la stratégie Contrariante Long Only, et pour nos 5 actifs séléctionnés (dont Sanofi), les résultats sont les suivants :

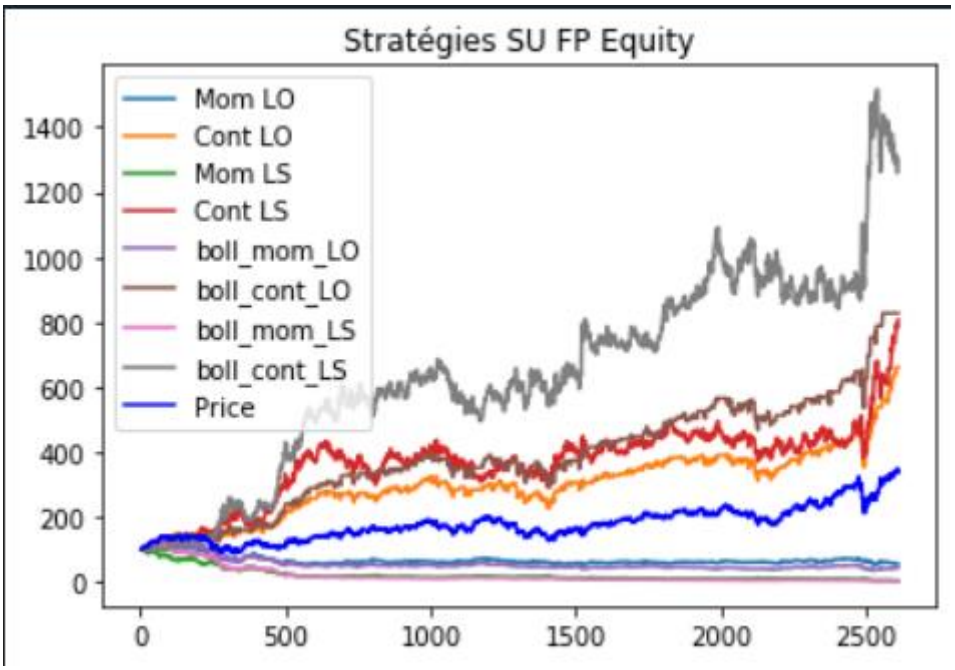
cont_LO - Series

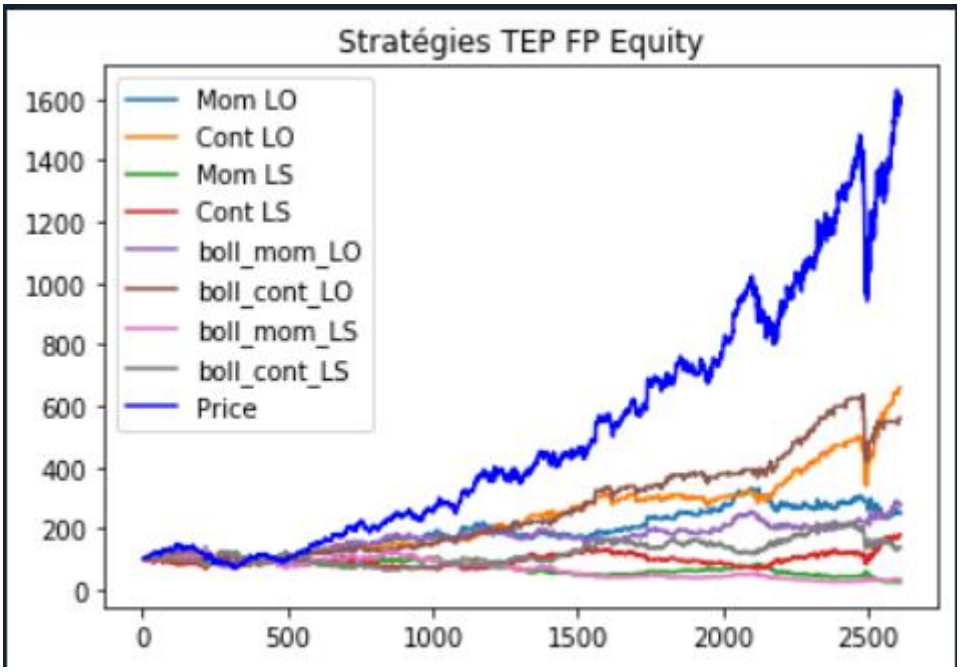
Indice	0
SAN FP Equity	13
SGO FP Equity	8
STM FP Equity	1
SU FP Equity	11
TEP FP Equity	11

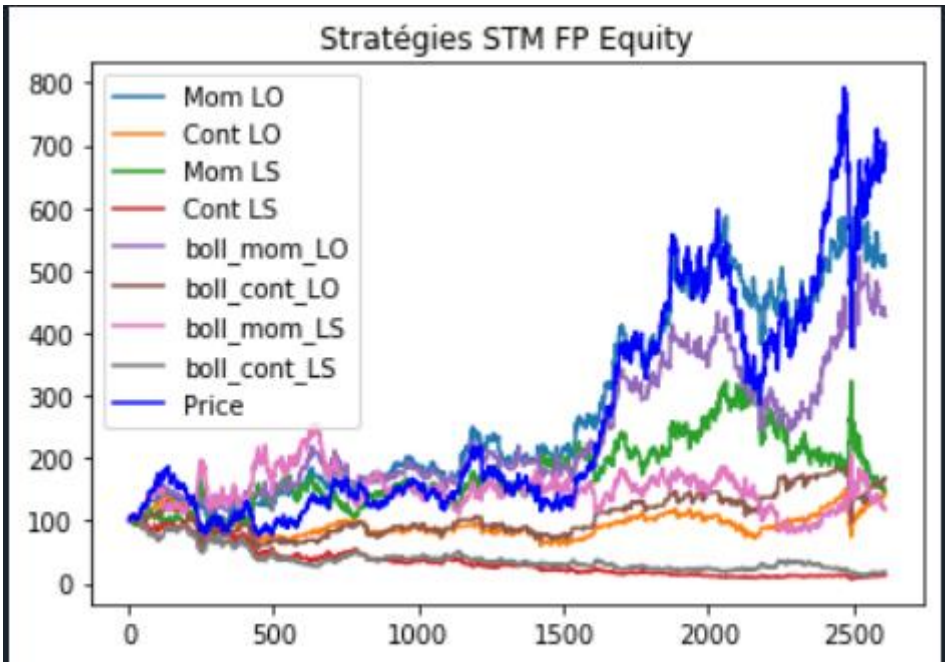
Pour n = 10, voyons à quoi ressemblent les stratégies :


Stratégies SAN FP Equity
Mom LO
Cont LO
Mom LS
Cont LS
boll_mom_LO
boll_cont_LO
boll_mom_LS
boll_cont_LS
Price
0
100
200
300
400
500
0
500
1000
1500
2000
2500


Stratégies SGO FP Equity
Mom LO
Cont LO
Mom LS
Cont LS
boll_mom_LO
boll_cont_LO
boll_mom_LS
boll_cont_LS
Price
0
200
400
600
800
0
500
1000
1500
2000
2500


Stratégies SU FP Equity
Mom LO
Cont LO
Mom LS
Cont LS
boll_mom_LO
boll_cont_LO
boll_mom_LS
boll_cont_LS
Price
1400
1200
1000
800
600
400
200
0
0
500
1000
1500
2000
2500


Stratégies TEP FP Equity
Mom LO
Cont LO
Mom LS
Cont LS
boll_mom_LO
boll_cont_LO
boll_mom_LS
boll_cont_LS
Price
1600
1400
1200
1000
800
600
400
200
0
0
500
1000
1500
2000
2500



Nous pouvons remarquer que pour les séries pour lesquelles le prix augmente fortement de lui-même sur la période, les stratégies de trading sont moins efficaces que l'investissement dans l'actif directement sur toute la période. Les stratégies sont le plus efficaces lorsque le titre est volatile et subit de fortes montées/ descentes.

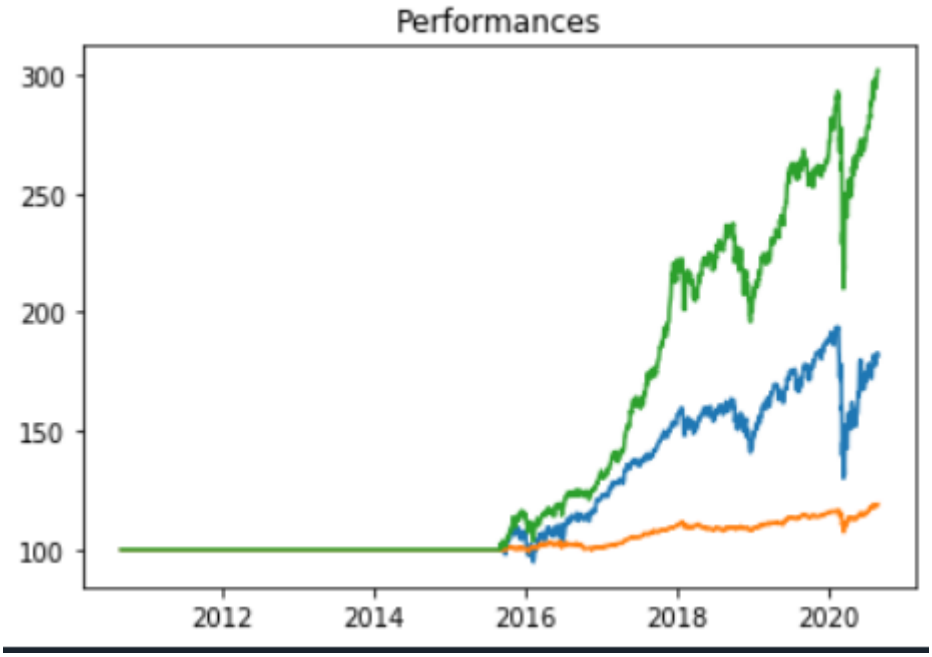
Remarque : le temps d'exécution du code est important parce qu'il s'agit de boucler sur les n, et sur les stratégies. En l'occurrence, les temps étaient très raisonnables sans le calcul de la stratégie de Bollinger.

Partie 2

Dans la partie 2, les DataFrame utilisés initialement ont été réindéxés de sorte à intégrer toutes les dates, y compris les jours non ouvrés. Ceci avait pour but de faciliter la manipulation des dates. La structure consiste d'abord à créer des DataFrame vides dans lesquels seront stockées les allocations pour chaque mois ainsi que les dates de la fenêtre glissante sur 5 ans. Il est à noter que je calcule des performances rebalancées et non de type « buy and hold ». Ensuite, par une boucle while qui s'arrête dès lors que la fenêtre dépasse la date limite de notre DataFrame, on fait glisser la fenêtre de mois en mois et on ajoute l'allocation optimale (selon l'objectif : min vol ou max sharpe) pour le mois qui suit, calculée à partir des données des 5 dernières années. La boucle se termine et on calcule ensuite la performance des portefeuilles (des 2 portefeuilles optimaux et d'un portefeuille équipondéré qui sert de référence).

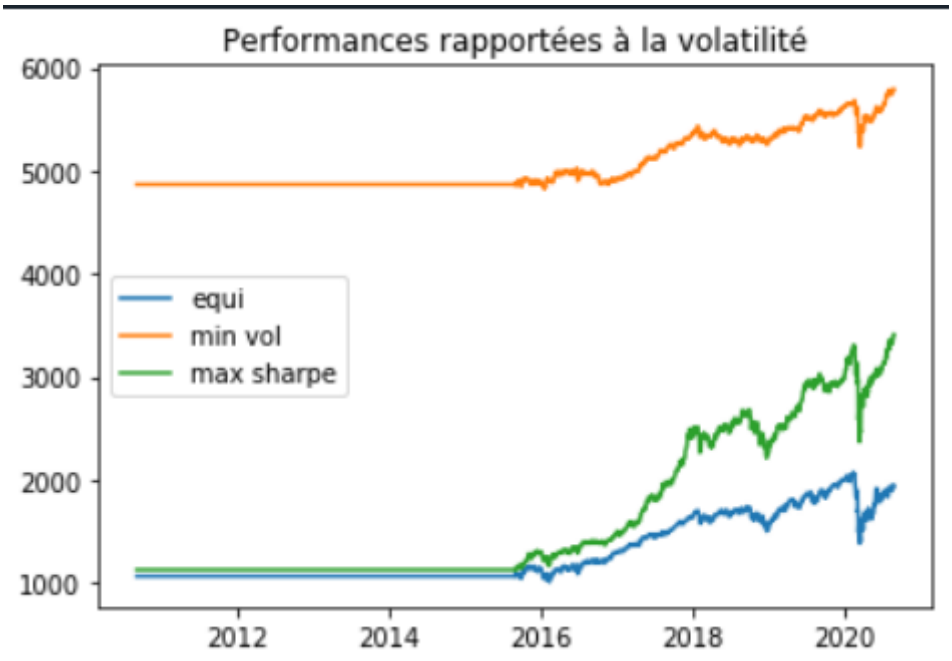
Les poids de chaque mois sont calculés à l'aide de la fonction GenerateWeights, qui tient compte de l'objectif d'optimisation. La fonction AddAlloc permet d'ajouter ces poids sur toutes les lignes du mois.

Le résultat que nous obtenons est le suivant :



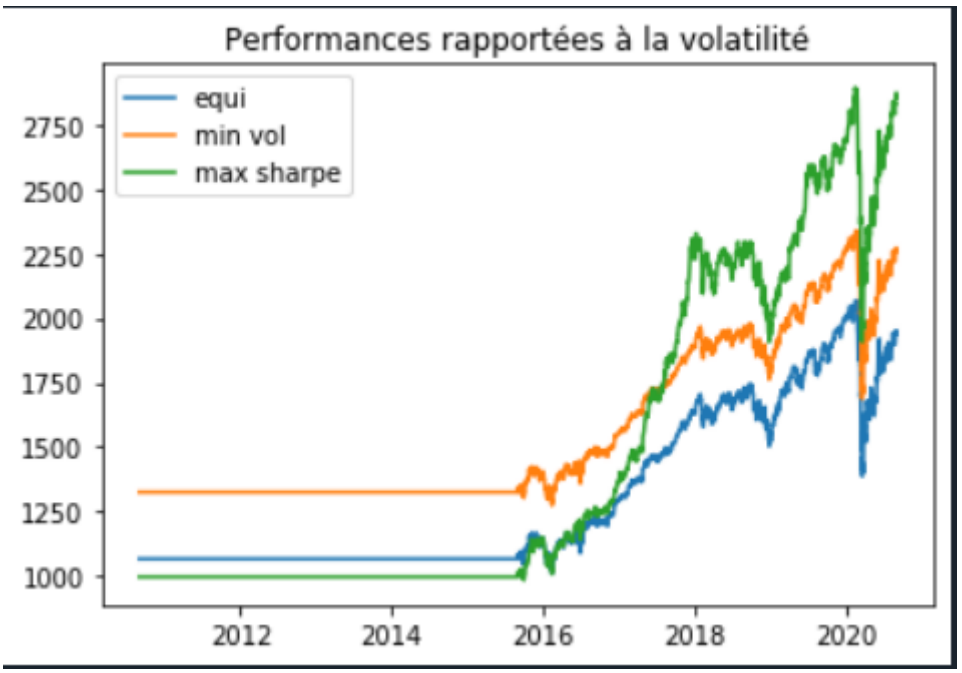
Nous pouvons constater que la stratégie max Sharpe est très efficace comparée à une allocation de type équipondérée. En revanche, l'allocation min vol nous permet de dégager que peu de performance en comparaison, elle représente le pari « sûr ».

Si l'on rapporte ces performances à la volatilité des 3 portefeuilles, on obtient :



Nous pouvons remarquer ici que performances rapportées à leur volatilité, les 3 stratégies ne se valent pas : la min vol est bien entendu la plus performante en ces termes. Nous pouvons également constater que le portefeuille équipondéré est sous-optimal : pour volatilité assez similaire à celle de l'allocation max sharpe, le rendement est 2 fois moins élevé. Les portefeuilles min vol et max sharpe sont donc les portefeuilles optimaux selon en fonction de l'aversion au risque de l'investisseur.

En termes d'amélioration du modèle, il est intéressant de le faire tourner en changeant les contraintes de poids sur les actifs et donc de mettre le poids minimum à 1% plutôt qu'à 0. En effet, les modèles d'optimisation ont tendance à concentrer les poids sur les actions ayant le meilleur rendement ou les moins volatiles (si on ignore l'impact des covariances pour simplifier l'explication), ce qui fait que l'on peut se retrouver short sur beaucoup d'actions. En mettant ce poids minimum, on oblige le solveur à trouver une solution optimale en possédant au minimum une très faible quantité de chaque actif.



En ce faisant, les performances sont évidemment amoindries mais le modèle rend mieux compte des enjeux de la gestion d'actifs : les sociétés de gestions ne peuvent se permettre de mettre leurs œufs dans le même panier, d'autant plus que c'est par la diversification que l'on peut réduire la volatilité et donc les risques. La performance du portefeuille max Sharpe était initialement de 200% sur les 10 ans et vaut environ 175% avec cette contrainte supplémentaire, ce qui reste une bonne performance.