AlgoInvest & Trade

Par Domnin BENOIT

Force Brute VS Optimized

Dans la présentation suivante, je vais vous présenter la comparaison entre l'algorithmes de force brute et la version optimisée.

Pour rappel de caractéristiques lié au projet:

- Un porte monnaie de 500 € par client
- Une action ne peut être acheter qu'une fois et en intégralité
- Un ensemble d'action avec le meilleur rendement doit ressortir

Nous avons plusieurs jeux de données:

- Un jeu pour force brute de 20 entrées
- 2 jeux avec environ 1000 entrées et avec certaines incohérences.

01 Description et Analyse

Détails des avantages et inconvénients de chaque option.

L'algorithme de force brute (bruteforce.py)

L'algorithme de force brute explore toutes les combinaisons possibles d'actions pour trouver la combinaison offrant le profit maximal. Pour chaque combinaison, il calcule le profit total en vérifiant chaque action individuelle et en calculant si elle doit être incluse ou non. Cela se fait par une approche récursive qui examine toutes les possibilités, ce qui peut être très coûteux en termes de temps et de ressources. La complexité temporelle de cet algorithme est exponentielle (O(2^n)) en fonction du nombre d'actions, ce qui peut rendre le traitement lent, surtout pour un grand nombre d'actions.

L'algorithme Optimisé (optimized.py)

Pseudocode:
Tri des actions par ratio profit/coût en ordre décroissant

Sélection des actions pour maximiser le profit :
  Pour chaque action dans la liste triée :
    Si le coût de l'action est inférieur ou égal à l'argent disponible :
      Ajouter l'action à la liste des actions sélectionnées
      Réduire l'argent disponible en conséquence

La complexité mémoire:

Elle reste linéaire O(n), où 'n' représente la taille des données d'entrée. Dans l'ensemble, cette approche semble être relativement efficace en termes d'utilisation de la mémoire, ce qui est important pour la gestion de jeux de données de différentes tailles.

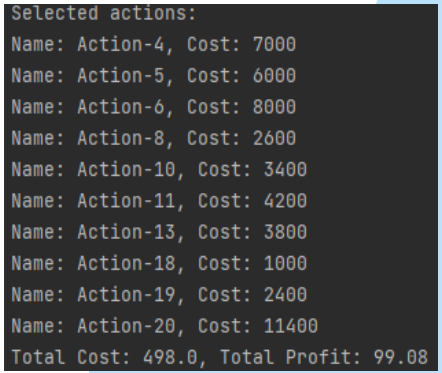
La complexité temporelle:

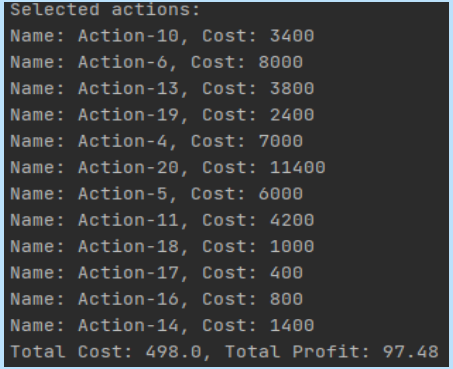
Elle est dominée par le tri initial en O(n log n) et le parcours des actions en O(n), ce qui nous donne une complexité globale de O(n log n). Cette approche heuristique, bien qu'efficace, diffère de la programmation dynamique utilisée dans l'algorithme classique du sac à dos, où les coûts et les valeurs sont combinés pour optimiser une fonction objectif tout en respectant des contraintes de poids.

Algorithme optimisé et ses limites

L'algorithme optimisé consiste à trier les actions en fonction du ratio profit/coût, puis à sélectionner les actions de manière gloutonne en fonction de ce tri. Cela permet de maximiser le profit en priorisant les actions les plus rentables par rapport à leur coût. Cependant, cette approche ne garantit pas toujours la solution optimale pour le problème du sac à dos. Dans certains cas, il peut donner une solution sous-optimale si le tri initial ne reflète pas parfaitement la solution optimale.

Force Brute



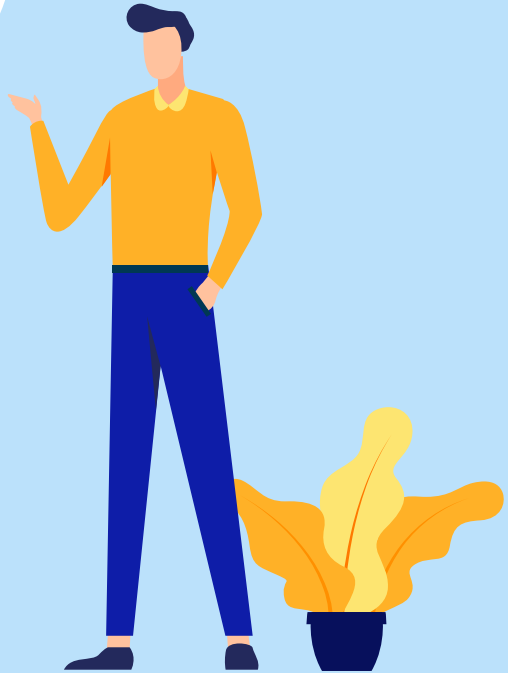


Optimisé

Conclusions

Sur un jeu de données limité, la solution de force brute permet un résultat optimale. Par contre des que les données commencent à être vaste, le délai est trop long et le programme ne retourne pas de résultats.

A l'inverse, la version optimisé permet même sur les jeux de données fournit un résultat quasiment instantané. Par contre la solution ne fournit pas le meilleur bénéfice.



02 Comparatif Sienna Algo Optimisé

Comparaison sur les 2 jeux de données testés pour voir la différence.

dataset1

```
Sienna bought:

Share-GRUT

Total cost: 498.76â,¬
Total return: 196.61â,¬
```

Sienna

Prix: 498,76

Bénéfice: 196,61

Nb actions achetées : 1

Rendement: 39,4%

Algo

Prix: 500

Bénéfice: 198,51

Nb actions achetées : 26

Rendement: 39,7%

```
Selected actions:
Name: Share-XJMO, Cost: 939
Name: Share-KMTG, Cost: 2321
Name: Share-MTLR, Cost: 1648
Name: Share-GTQK, Cost: 1540
Name: Share-LRBZ, Cost: 3290
Name: Share-WPLI, Cost: 3464
Name: Share-GIAJ, Cost: 1075
Name: Share-GHIZ, Cost: 2800
Name: Share-IFCP, Cost: 2923
Name: Share-ZSDE, Cost: 1511
Name: Share-FKJW, Cost: 2108
Name: Share-NHWA, Cost: 2918
Name: Share-LPDM, Cost: 3935
Name: Share-QQTU, Cost: 3319
Name: Share-USSR, Cost: 2562
Name: Share-EMOV, Cost: 889
Name: Share-LGWG, Cost: 3141
Name: Share-SKKC, Cost: 2487
Name: Share-QLMK, Cost: 1738
Name: Share-UEZB, Cost: 2487
Name: Share-CBNY, Cost: 122
Name: Share-CGJM, Cost: 1721
Name: Share-EVUW, Cost: 444
Name: Share-FHZN, Cost: 610
Name: Share-MLGM, Cost: 1
Name: Share-DBUJ, Cost: 7
Total Cost: 500.0, Total Profit: 198.5101
```

dataset2

```
Sienna bought:
Share-ECAQ 3166
Share-IXCI 2632
Share-FWBE 1830
Share-ZOFA 2532
Share-PLLK 1994
Share-YFVZ 2255
Share-ANFX 3854
Share-PATS 2770
Share-NDKR 3306
Share-ALIY 2908
Share-JWGF 4869
Share-JGTW 3529
Share-FAPS 3257
Share-VCAX 2742
Share-LFXB 1483
Share-DWSK 2949
Share-XQII 1342
Share-ROOM 1506

Total cost: 489.24â,¬
Profit: 193.78â,¬
```

Sienna

Prix: 489,24

Bénéfice: 193,78

Nb actions achetées : 18

Rendement: 39,5%

Algo

Prix: 499,96

Bénéfice: 197,76

Nb actions achetées : 22

Rendement: 39,56%

```
Selected actions:
Name: Share-PATS, Cost: 2770
Name: Share-JWGF, Cost: 4869
Name: Share-ALIY, Cost: 2908
Name: Share-NDKR, Cost: 3306
Name: Share-PLLK, Cost: 1994
Name: Share-FWBE, Cost: 1830
Name: Share-LFXB, Cost: 1483
Name: Share-ZOFA, Cost: 2532
Name: Share-ANFX, Cost: 3854
Name: Share-FAPS, Cost: 3257
Name: Share-LXZU, Cost: 424
Name: Share-XQII, Cost: 1342
Name: Share-ECAQ, Cost: 3166
Name: Share-JGTW, Cost: 3529
Name: Share-IXCI, Cost: 2632
Name: Share-DWSK, Cost: 2949
Name: Share-ROOM, Cost: 1506
Name: Share-VCXT, Cost: 2919
Name: Share-YFVZ, Cost: 2255
Name: Share-OCKK, Cost: 316
Name: Share-JMLZ, Cost: 127
Name: Share-DYVD, Cost: 28
Total Cost: 499.96, Total Profit: 197.7593
```