Spécifications techniques

Partie 1 : solution de brut force

Fichier («bruteforce.py»)

Concevoir un algorithme qui maximisera le profit réalisé par les clients **après 2 ans d’investissement**.

L’algorithme doit suggérer une liste des actions les plus rentables que l’entreprise doit acheter pour maximiser le profit d’un client au bout de deux ans. Le programme doit lire un fichier contenant des informations sur les actions, essayer toutes les différents combinaisons d’actions qui correspondent aux contraintes. Il doit retourner le meilleur résultat.

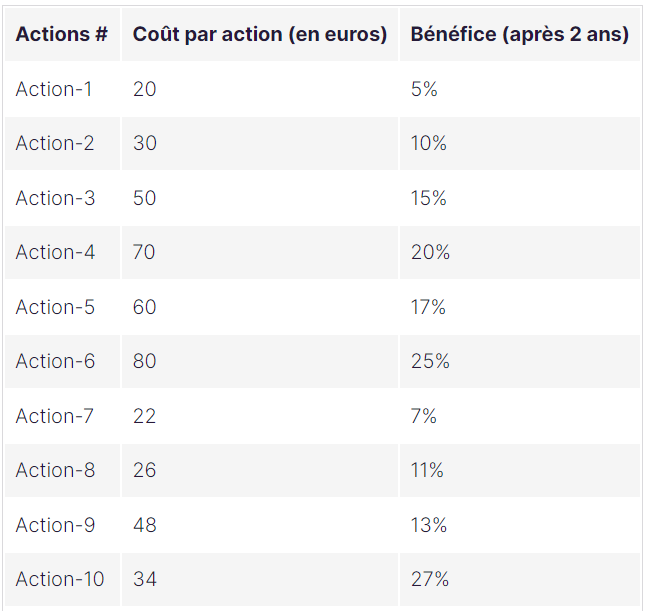
Contraintes:

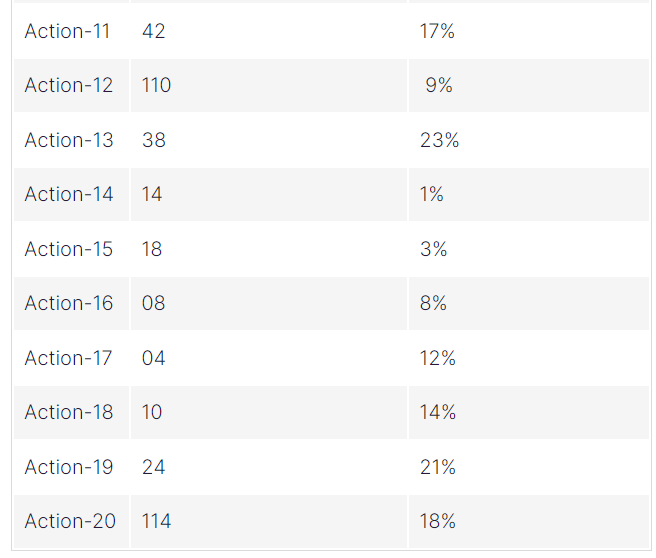
- **chaque action ne peut être achetée qu’une seule fois**

- l’achat de fraction d’action n’est pas possible

- **budget maximum de 500€ par client**

Liste des actions disponibles :





Partie 2: Optimisation d’algorithme

Fichier («optimized.py»)

La première version du programme fonctionne mais prend trop de temps. La deuxième version devra fournir une réponse en moins de **1 seconde**. Plus besoin d’explorer toutes les combinaisons possibles. Pour cette deuxième partie, il faut présenter :

- un programme optimisé, qui lit un fichier contenant des informations sur les actions et fournit la meilleure version d’une stratégie

- un jeu de diapositive contenant les éléments suivants :

- une analyse de l’algorithme de force brute

- un diagramme, un organigramme ou un pseudo code décrivant les processus de réflexion qui sous-tend la solution optimisée

- l’agorithme choisi pour la version optimisée, les limites de l’algorithme (cas limites)

- une comparaison de l’efficacité et des performances de l’algorithme de force brute par rapport à l’algorithme optimisé en utilisant la notation Big-O, la complexité temporelle et l’analyse de la mémoire.

Partie 3: backtesting et optimisation

A ce stade, il faut tester l’exactitude du programme en l’exécutant sur des ensembles de données antérieurs et en vérifiant les résultats de l’algorithme. Les données et les décisions d’investissement correspondantes sont disponibles en ligne (il y a des données manquantes et incorrectes...)

Il faut produire un jeu de diapositives contenant une comparaison entre les résultats de l’algorithme et le choix du conseiller financier, ainsi qu’un rapport d’exploration des données.

Liste des livrables :

1. Un fichier python ‘bruteforce.py’
2. Un fichier python ‘optimized.py’
3. Un jeu de diapositives au format PDF (20 diapos maximum):

- la solution optimisée :

- l’analyse de l’algorithme de force brute

- un diagramme/agenda/pseudo code

- l’algorithme choisi et ses limites

- l’analyse des performances et de l’efficacité des algorithmes écrit

- une comparaison côte à côte entre la sortie de algorithme écrit et les choix du conseiller financier, via un rapport d’exploration de l’ensemble des données

Ressources :

- introduction à la O(N) notation

- Notation Big O

- Article : All You Need to Know About Dynamic Programming

- Livres : Problem Solving with Algorithms and Data Structures using Python

- cours : préparez-vous aux test d’algorithmique