COCOTIER VOLUME

Il y aura une partie back test et une partie live

- Back test :

La partie BACK TEST permettra d'analyser la pertinence de l'algorythme dans le passé même si les performances du passé ne garantissent pas les performances du futur.

Il faudra aussi étudier quel est le meilleur jeu de paramètre

- Live :

Le live permettra de mettre en application ce qui a été réalisé dans le back test en mettant en input le meilleur jeu de paramètre mis en evidence dans la partie back test

- Explication Back test
    - Explication macro :

Le but de l'algoryhtme est de choisir la Crypto qui a la meilleur evolution de volume (sur une période défini) parmi toutes les crypto de binance et ensuite d'appliquer le bot cocotier

    - Les différents variables :

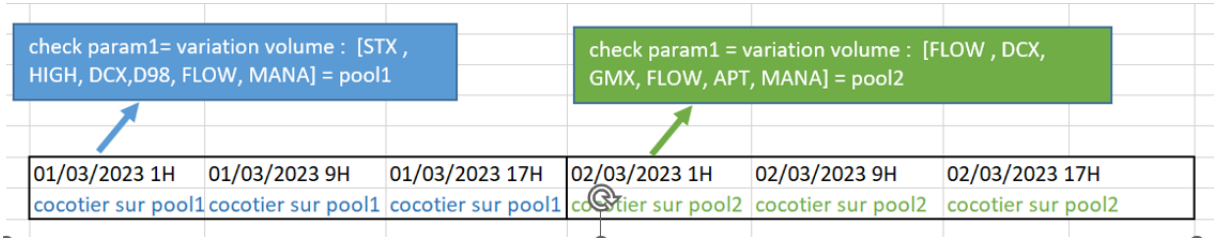
Variable 1 : il s'agira de la période à laquelle on effectue une observation de la variation de volume (6H/12H/24H…) si on choisit 12h il faudra prendre en compte la variation des 12 dernières heures

Variable 2 : la pool crypto, il s'agira du nombre de crypto qui seront dans la pool

Variable 3 : période à laquelle on va acheter et vendre la crypto (4H/6H/8H/12h)

Il s'agit de la « plage hauraire » qui est dans le back test cocotier

Exemple :



Dans ce cas :

variable 1 = 24h car on effectue l'observation à 1H le 1/03 et à 1H le 2/03

variable 2 = 6 car il y a 6 crypto dans la pool

variable 3 = ici = 8 car on effectue achat vente à 1h / 9h et 17h

- récupération de l'evolution du volume :

il y a 2 étapes

  - premiere étape : on recupere la data brut

Le script « crypto robot » permet de récupérer la data des 1000 dernières heures pour chaque crypto (= 41,5) et ensuite d'analyser l'evolution :

Récupération data

vidéo youtube : https://youtu.be/lbNzbktQeP0

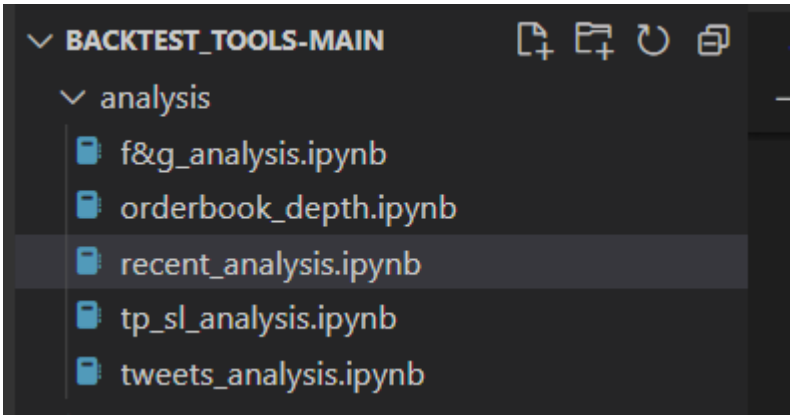
voici le repo github : https://github.com/CryptoRobotFr/backtest_tools

Pour recupere la data, il faut lancer un terminal et taper node .\database\sdl_for_quick_analysis.js

```
Run `npm audit` for details.
PS C:\Users\aniss\Desktop\crpt\CODE_CRYTPOROBOT\backtest_tools-main> node .\database\download_data.js
BTC/USDT Binance 1h 01-06-2017
Loading 44/44 requests | 43028 candles loadedETH/USDT Binance 1h 01-06-2017
Loading 44/44 requests | 43028 candles loadedLINK/USDT Binance 1h 01-06-2017nce/1h/BTC-USDT.csv
Loading 44/44 requests | 43028 candles loadedDOGE/USDT Binance 1h 01-06-2017nce/1h/ETH-USDT.csv
Loading 44/44 requests | 43028 candles loadedSOL/USDT Binance 1h 01-06-2017nance/1h/LINK-USDT.csv
Loading 44/44 requests | 43028 candles loadedXEM/USDT Binance 1h 01-06-2017ance/1h/DOGE-USDT.csv
Successfully downloaded 12469 candles since 24-11-2020 13:0 in ./database/Binance/1h/XEM-USDT.csv
PS C:\Users\aniss\Desktop\crpt\CODE_CRYTPOROBOT\backtest_tools-main>
```

  - deuxième étape : on organise la data dans un dataframe et on analyse

pour réaliser cela c'est au niveau du code « recent analysis »



Dans le code ici on a choisi une fenetre de 24h (= variable1) et on a filtre par l'evolution du volume (volume evolution)

```
df_metric = get_analyisis_from_window(24).sort_values(by="volume_evolution", ascending=False)
df_metric.iloc[:30]
```

✓ 2.8s

	mean_volume	last_volume	volume_evolution	mean_volatility	last_volatility	volatility_evolution	return	last_return	return_vs_vol	last_return_vs_vol
LRC	1486186	14914249	10.04	1.90	16.24	8.53	0.76	31.33	0.40	1.93
ASR	192667	1292466	6.71	2.29	4.31	1.88	17.26	-9.86	7.55	-2.29
SRM	450719	2732464	6.06	1.55	5.98	3.85	9.98	13.29	6.43	2.22
BTG	38023	224562	5.91	1.42	3.85	2.71	22.81	9.35	16.10	2.43
OXT	161476	893607	5.53	1.74	5.05	2.89	-9.02	5.81	-5.17	1.15
TWT	499824	2519308	5.04	2.31	4.21	1.83	67.89	-4.90	29.42	-1.16
ETC	2844401	14245427	5.01	1.70	3.48	2.05	31.35	9.23	18.46	2.65
DAR	831445	3943634	4.74	2.52	2.94	1.17	-19.97	-3.63	-7.93	-1.23
IOTX	739034	2859605	3.87	1.88	1.90	1.01	-8.83	4.59	-4.70	2.42
FET	414417	1411942	3.41	2.31	4.43	1.92	-1.96	4.28	-0.85	0.97
IASMY	3973068	12541956	3.16	3.92	4.67	1.19	10.51	4.87	2.68	1.04

[Mean volume] = volume moyen les 1000 derniere heures
[Last volume] = volume moyen les 24 derniers heures
[volume evolution] ICI LRC a 10 fois plus de volume que d habitude sur les **24 derniers heures** par rapport au 1000 dernieres heures
[Last volatility] = analyse de la volatilité 24 derniers heures
[Volatility evolution] ceux qui ont le plus eu d"evolution sur leur volatilité
[Return] rendement ceux qui ont le mieux performé sur les 1000 dernieres heures
[Last return] rendement ceux qui ont le mieux performé sur notre periode à savoir la window = 24 dernieres heures
[Return vs vol] ceux qui ont le mieux performé vs la volatilité sur les 1000 dernieres heures (le but est d avoir le moins de volatilité et un meilleur retour c est à dire pas des pics en dent de scie, ici on a une tendance des crypto qui ont performé sur le moyen terme
[Last return vs vol] la meme mais cette fois ci sur les 24 dernieres heures

Pour notre projet, ce qui nous interesse c'est [volume evolution]
Ici **volume evolution = last volume / mean volume** => on realise la comparaison du volume des 24 dernieres heures par rapport au volume moyen les 1000 dernieres heures

Contrainte technique

- Langage : python
- le backtest se fera sur un notebook
- crée un nouveau repositorie github :
    - le code doit etre transportable sur un environnement indépendant
        - environnement independant (.env) mais pas affilier au reposetorie
    - fichier git :
        - Readme.md
        - requierement.txt
        - .gitignore
        - BackTestVolume.ipynb
        - autre dossier ou fichier utilitaire au projet