

Traitement de données - Analyse de prédition de marchés financiers

1. Exploration, compréhension et nettoyage

Une rapide observation des données à permis de lister les types attendu pour chaque colonne. Il a alors été possible de créer des scripts pour chaque dataset et transformer les données pour faciliter le traitement par la suite.

Il a ensuite été possible de créer une fonction qui permet d'optimiser les colonnes numérique afin de réduire leur taille et ainsi optimiser l'import et les traitement futur.

Une chose plus compliqué dans les données était le format de date du fichier sto.parquet, en effet ses dates étaient sous la forme d'un nombre relativement petit qui ne correspondant donc pas à un timestamp.

J'ai alors supposé que la colonne affichait le nombre de jour écoulé depuis une certaine date.

Au final j'ai obtenu ce type de donnée :

Pour le cac40.csv :

```
==== CAC40 ====
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 533 entries, 0 to 532
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype  
--- 
 0   Date     533 non-null    datetime64[ns]
 1   Index    533 non-null    category
 2   Open     527 non-null    float32 
 3   High     520 non-null    float32 
 4   Low      529 non-null    float32 
 5   Close    529 non-null    float32 
 6   Volume   523 non-null    float32 
dtypes: category(1), datetime64[ns](1), float32(5)
memory usage: 15.3 KB
```

Pour le sto.parquet

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 657 entries, 0 to 656
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype  
--- 
 0   Date     657 non-null    datetime64[ns]
 1   Close    657 non-null    float32 
 2   High     657 non-null    float32 
 3   Low      657 non-null    float32 
 4   Open     657 non-null    float32 
 5   Volume   657 non-null    int32  
dtypes: datetime64[ns](1), float32(4), int32(1)
```

J'ai pu ensuite faire différent calcul comme :

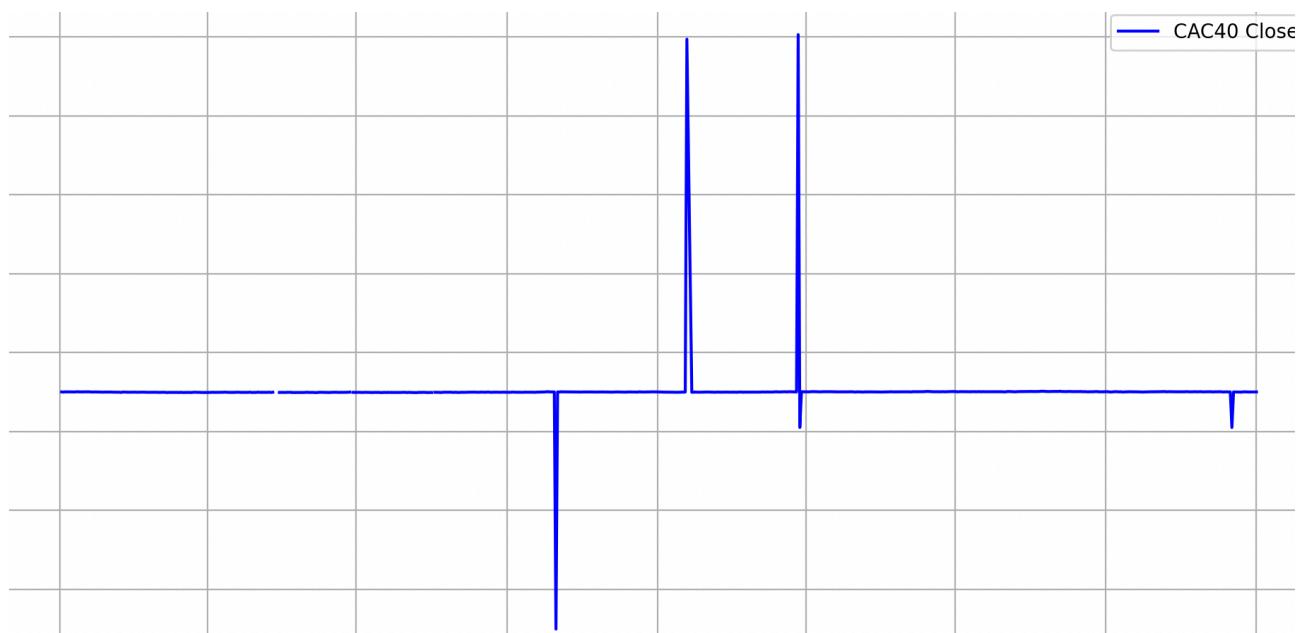
- Les ranges de date des données :

```
cac40: 2023-01-02 00:00:00 => 2025-01-01 00:00:00  
sto: 2023-01-01 00:00:00 => 2025-10-13 00:00:00
```

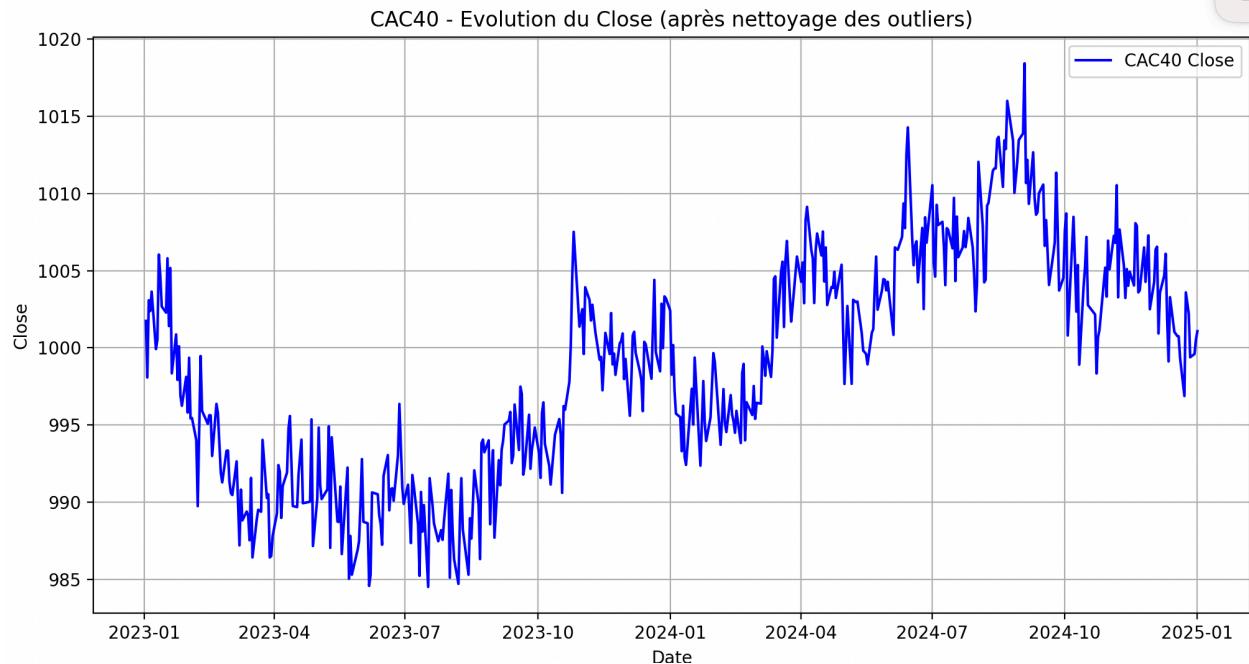
- Les quantités de valeurs manquantes pour chaque colonne de chaque dataset

```
cac40:  
Date      0  
Index     0  
Open      6  
High      13  
Low       4  
Close     4  
Volume    10  
dtype: int64  
  
sto:  
Date      0  
Close     0  
High      0  
Low       0  
Open      0  
Volume    0  
dtype: int64
```

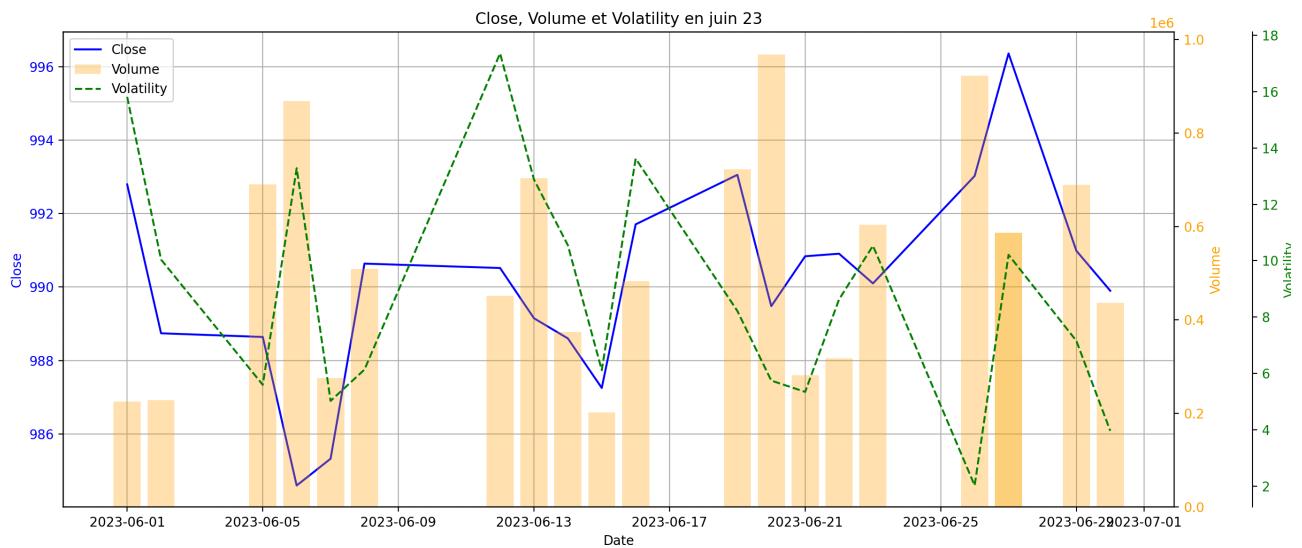
Le dataset contenait également quelque valeur absurde qui faussait le résultat.
Il est alors impossible de voir une quelconque évolution avec ceci. J'ai utilisé une méthode avec les écart types pour filtrer les données et enlever les données irréalistes.



Les valeurs Close sont alors bien plus lisibles et cohérentes avec la réalité.

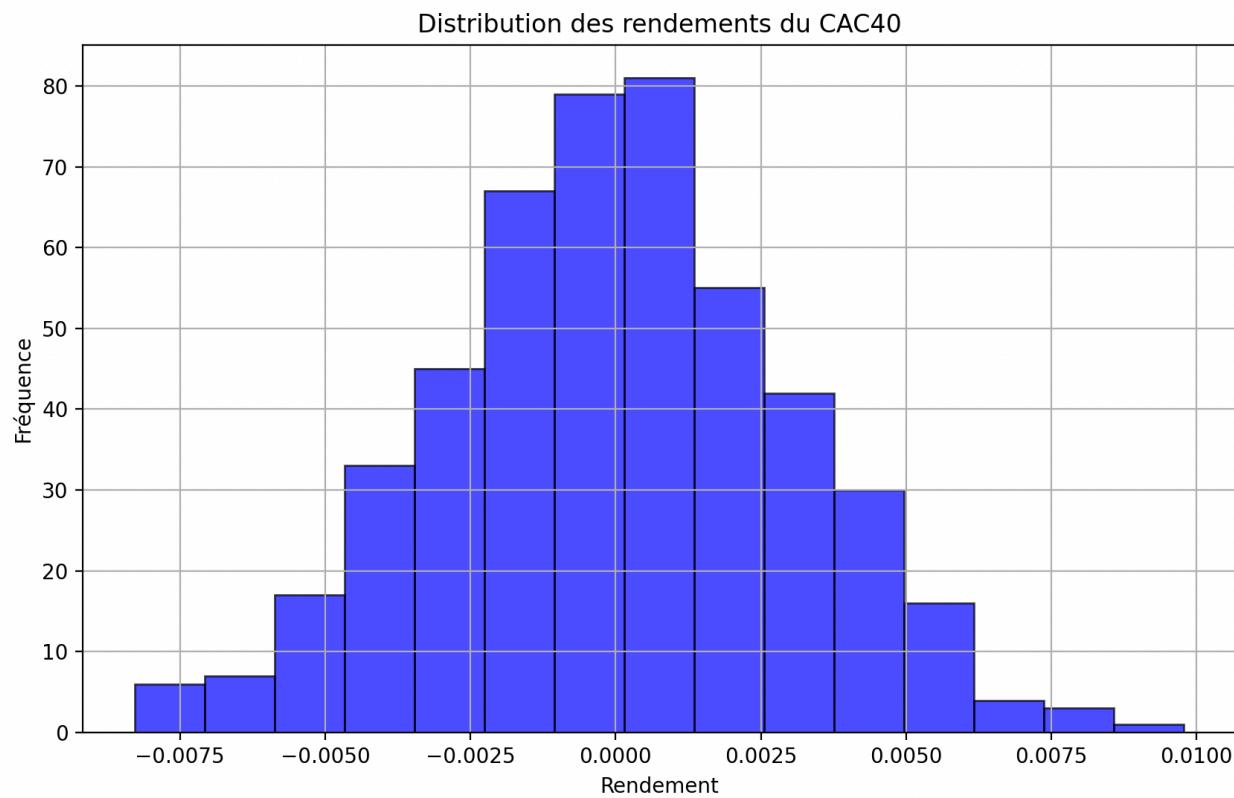


En superposant la volatility (high - low), le close et le volume on peut obtenir ce graph qui est zoomé sur 1 mois (juin 23) pour avoir une meilleure visibilité



Les rendements peuvent être calculé avec une formule logarithmique :
Rendement n = $\ln(\text{prixCloture } n / \text{prixCloture } n-1)$

On obtient cet histogramme :



2. Phase d'analyse statistique

Dans cette partie l'objectif est de commencer un petit modèle qui serait capable à partir de l'étude de certains indicateur précis, de prédire si le cours va augmenter ou diminuer.

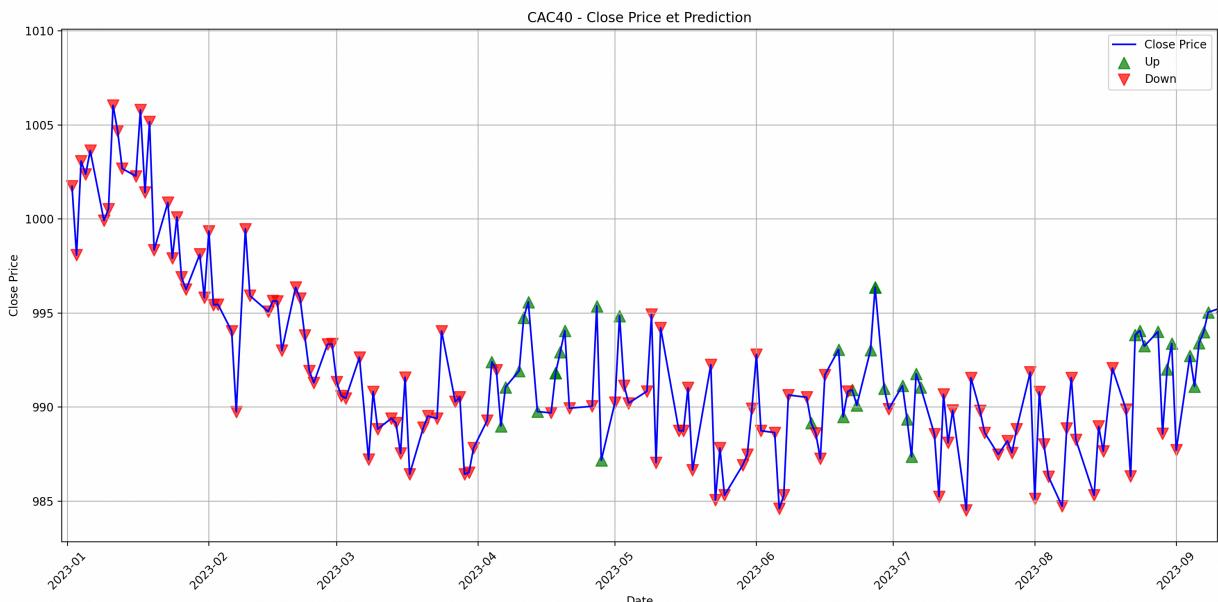
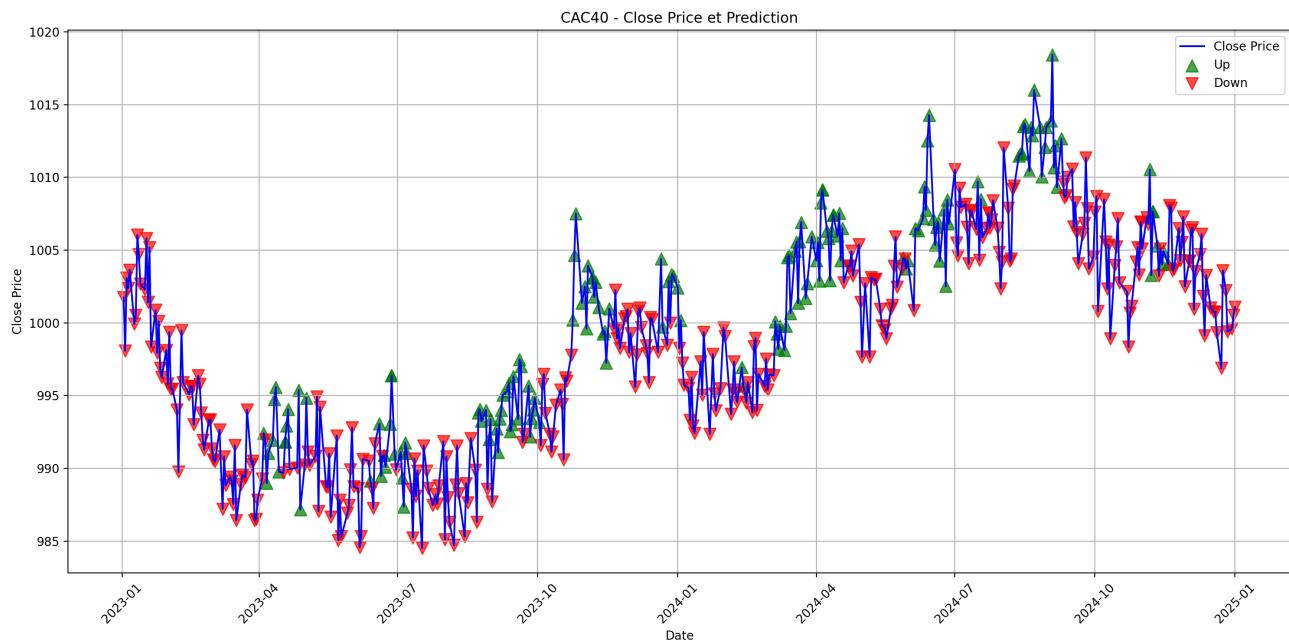
Le choix a été fait de se baser sur différentes moyennes mobile et sur le momentum.

J'ai alors créer une fonction qui permet de calculer ces valeurs sur le data frame et qui enregistre ensuite ces modifications dans un fichier .parquet.

Par la suite, en utilisant la règle de statistique données, j'ai pu en déduire les moment où le cours devrait monter et quand il devrait baisser.

J'ai pu obtenir un résultat de précision de 0.56 (56.26%)

J'ai ensuite pu visualiser cela via un graphique qui met en parallèle les valeurs réelles et les valeurs prévues.



Il est ensuite possible d'étendre l'apprentissage en utilisant un modèle comme RandomForestClassifie, en faisant cela en calculant tout un tas de paramètre donnée, En testant les 20% dernière données, j'obtiens un taux d'accuracy à 60,64%

accuracy : 0.56 (56.26%)				
Accuracy du modèle RandomForestClassifier: 60.64%				
	precision	recall	f1-score	support
0	0.57	0.71	0.63	45
1	0.66	0.51	0.57	49
accuracy			0.61	94
macro avg	0.61	0.61	0.60	94
weighted avg	0.62	0.61	0.60	94