Conversation APP bourse

Est-ce qu'on peut créer un nouveau modèle qui va nous permet de dire quand est-ce qu'une action va atteindre son pic dans le mois afin de peut être chercher à revendre Et quand est-ce qu'il va atteindre son down dans le mois afin de peut être chercher à acheter

###### ChatGPT said:

Oui, il est tout à fait possible de créer un modèle qui tente de prédire quand une action atteindra son pic ou son plus bas (down) dans le mois. Cependant, il faut comprendre que la prévision précise des points hauts (pics) et bas (creux) dans un marché est très difficile à réaliser de manière précise et fiable, car les marchés financiers sont influencés par une multitude de facteurs, dont certains sont imprévisibles.

Cela dit, vous pouvez mettre en place une approche basée sur des techniques statistiques, des modèles de prévision et des indicateurs techniques pour vous aider à estimer les moments où une action pourrait atteindre un pic ou un creux. Voici quelques idées pour structurer un tel modèle.

### 1. Approche Basée sur des Indicateurs Techniques

Les indicateurs techniques sont largement utilisés pour analyser les tendances des prix d'une action. Voici les indicateurs principaux qui pourraient vous aider à identifier les pics et les creux dans le prix d'une action :

* Bandes de Bollinger : Cet indicateur peut vous aider à repérer les niveaux de surachat et de survente. Lorsqu'une action dépasse la bande supérieure, elle est considérée comme surachetée (potentiel pic), tandis que lorsqu'elle tombe sous la bande inférieure, elle est considérée comme survendue (potentiel bas).
* RSI (Relative Strength Index) : Un RSI élevé (généralement au-dessus de 70) indique que l'action est surachetée (possible pic), tandis qu'un RSI faible (généralement en dessous de 30) indique que l'action est survendue (possible creux).
* Moyenne mobile (SMA ou EMA) : Les croisements de moyennes mobiles (par exemple, croisement de la moyenne mobile à court terme au-dessus de la moyenne mobile à long terme) peuvent signaler un point d'inversion de tendance.
* MACD (Moving Average Convergence Divergence) : Cet indicateur est utilisé pour identifier les points de retournement des tendances, en particulier les croisements de MACD et de la ligne de signal.

### 2. Approche Statistique : Modèle Basé sur la Série Temporelle

Un modèle de série temporelle peut être utilisé pour prédire les mouvements de prix d'une action et détecter les moments où elle atteint un pic ou un creux. Vous pourriez utiliser :

* Modèles ARIMA (AutoRegressive Integrated Moving Average) : Ces modèles sont efficaces pour prédire les valeurs futures en fonction des valeurs passées. Vous pouvez utiliser ARIMA pour prévoir les futurs pics ou creux.
* Modèles GARCH : Utilisés pour prédire la volatilité future des prix, ils peuvent vous aider à identifier les périodes de volatilité élevée ou faible, qui pourraient être liées aux pics ou creux.

### 3. Création du Modèle : Exemple Utilisant des Indicateurs Techniques

Je vais vous montrer un exemple de code utilisant les Bandes de Bollinger et l'RSI pour identifier des pics et des creux potentiels.

### Étape 1 : Calcul des Bandes de Bollinger et de l'RSI

Voici un exemple de code en Python utilisant des indicateurs techniques pour tenter de prévoir les pics et les creux.

python

Copy code

import pandas as pd

import numpy as np

import talib

# Suppose que price\_df est un DataFrame avec les prix de clôture des actions

price\_df = pd.DataFrame({

'Date': pd.date\_range(start='2023-01-01', periods=100),

'Close': np.random.randn(100).cumsum() + 100

})

# --- Bandes de Bollinger ---

def calculate\_bollinger\_bands(prices, window=20):

"""

Calcule les bandes de Bollinger pour un ensemble de prix.

Parameters:

prices (pd.Series): Prix de clôture

window (int): Période pour calculer la moyenne mobile et les bandes

Returns:

pd.DataFrame: Moyenne mobile, bande supérieure et bande inférieure

"""

rolling\_mean = prices.rolling(window).mean()

rolling\_std = prices.rolling(window).std()

upper\_band = rolling\_mean + (rolling\_std \* 2)

lower\_band = rolling\_mean - (rolling\_std \* 2)

return rolling\_mean, upper\_band, lower\_band

# Appliquer la fonction

rolling\_mean, upper\_band, lower\_band = calculate\_bollinger\_bands(price\_df['Close'])

# --- RSI (Relative Strength Index) ---

def calculate\_rsi(prices, window=14):

"""

Calcule le RSI (Relative Strength Index) pour un ensemble de prix.

Parameters:

prices (pd.Series): Prix de clôture

window (int): Période pour calculer le RSI

Returns:

pd.Series: RSI

"""

return talib.RSI(prices, timeperiod=window)

# Appliquer le RSI

rsi = calculate\_rsi(price\_df['Close'])

# --- Identification des pics et creux ---

def identify\_peaks\_and\_troughs(prices, rsi, upper\_band, lower\_band):

"""

Identifie les pics (points de surachat) et les creux (points de survente).

Parameters:

prices (pd.Series): Prix de clôture

rsi (pd.Series): Valeurs du RSI

upper\_band (pd.Series): Bande supérieure des Bandes de Bollinger

lower\_band (pd.Series): Bande inférieure des Bandes de Bollinger

Returns:

pd.DataFrame: Points d'achat et de vente potentiels (creux et pics)

"""

signals = pd.DataFrame(index=prices.index)

signals['Price'] = prices

signals['RSI'] = rsi

signals['Upper\_Band'] = upper\_band

signals['Lower\_Band'] = lower\_band

signals['Signal'] = 0 # 0 pour aucun signal, 1 pour vendre, -1 pour acheter

# Condition pour vendre (possible pic) : RSI > 70 et prix dépasse la bande supérieure

signals.loc[(signals['RSI'] > 70) & (signals['Price'] > signals['Upper\_Band']), 'Signal'] = 1

# Condition pour acheter (possible creux) : RSI < 30 et prix tombe sous la bande inférieure

signals.loc[(signals['RSI'] < 30) & (signals['Price'] < signals['Lower\_Band']), 'Signal'] = -1

return signals

# Appliquer la fonction d'identification

signals = identify\_peaks\_and\_troughs(price\_df['Close'], rsi, upper\_band, lower\_band)

# Affichage des points d'achat et de vente potentiels

print(signals[signals['Signal'] != 0])

### Explication du Code

1. Bandes de Bollinger :
   * La fonction calculate\_bollinger\_bands calcule les bandes de Bollinger basées sur une période glissante (par défaut 20 jours). Les bandes indiquent les niveaux de surachat (bande supérieure) et de survente (bande inférieure).
2. RSI (Relative Strength Index) :
   * La fonction calculate\_rsi calcule l'RSI qui permet de déterminer si une action est surachetée ou survendue. Lorsque l'RSI dépasse 70, cela indique une action surachetée (potentiel pic), et lorsqu'il descend sous 30, cela indique une action survendue (potentiel creux).
3. Identification des Pics et Creux :
   * La fonction identify\_peaks\_and\_troughs génère des signaux d'achat et de vente. Un signal d'achat est généré lorsque le prix est sous la bande inférieure et que le RSI est inférieur à 30 (possible creux). Un signal de vente est généré lorsque le prix est au-dessus de la bande supérieure et que le RSI est supérieur à 70 (possible pic).

### Étape 2 : Amélioration du Modèle

Vous pouvez améliorer ce modèle en combinant plusieurs indicateurs techniques et en ajustant les périodes et paramètres des indicateurs pour mieux capturer les cycles de votre action.

### Étape 3 : Modèle Basé sur une Série Temporelle

En parallèle, vous pouvez utiliser un modèle de série temporelle comme ARIMA pour faire des prévisions sur les prix futurs d'une action. Cela pourrait être couplé avec les indicateurs techniques pour affiner les prédictions sur les pics et creux.

### Conclusion

Bien que prédire précisément les pics et creux soit extrêmement difficile, un modèle basé sur les indicateurs techniques comme les Bandes de Bollinger, l'RSI, et les moyennes mobiles peut vous aider à repérer des moments où une action est probablement surachetée ou survendue, ce qui peut vous guider dans vos décisions d'achat et de vente. Vous pouvez également coupler cette approche avec un modèle de prévision statistique pour améliorer la fiabilité des prédictions.

1/2

4o

##### You said:

Développe l’approche avec ARIMA

###### ChatGPT said:

L'approche avec ARIMA (AutoRegressive Integrated Moving Average) est une méthode couramment utilisée pour modéliser des séries temporelles et faire des prévisions. ARIMA peut être utilisée pour modéliser les prix historiques d'une action et essayer de prédire les futurs points hauts (pics) ou points bas (creux) en analysant la tendance des prix.

### Comprendre ARIMA

ARIMA est un modèle de séries temporelles qui combine trois composants :

* AR (AutoRegressive) : Le modèle prend en compte les valeurs passées (retards) de la série pour faire des prévisions. La partie AR est contrôlée par le paramètre p, qui indique combien de valeurs passées sont utilisées.
* I (Integrated) : Ce composant rend la série temporelle stationnaire en prenant la différence entre les valeurs d'une période et de la période précédente. Il est contrôlé par le paramètre d, qui indique combien de fois les différences sont appliquées.
* MA (Moving Average) : Ce composant prend en compte les erreurs passées pour améliorer les prévisions. La partie MA est contrôlée par le paramètre q, qui indique combien d'erreurs passées sont utilisées.

L'objectif d'ARIMA est de faire des prévisions basées sur les prix historiques. Si les prévisions montrent que le prix d'une action est susceptible d'augmenter rapidement (potentiel pic) ou de baisser (potentiel creux), vous pouvez alors ajuster vos stratégies d'achat et de vente.

### Étapes pour Utiliser ARIMA pour Identifier les Pics et Creux

1. Modéliser la Série Temporelle : Utiliser le modèle ARIMA pour modéliser les prix de clôture d'une action.
2. Prévoir les Prix Futurs : Prédire les prix futurs de l'action en fonction des prix passés.
3. Identifier les Pics et Creux : Analyser les prévisions pour repérer les pics et creux dans le mois.

### Exemple de Modélisation avec ARIMA en Python

Voici comment vous pouvez utiliser le modèle ARIMA avec la bibliothèque statsmodels pour prédire les prix futurs d'une action.

#### Installation des Librairies

bash

Copy code

pip install pandas numpy statsmodels matplotlib

#### Étape 1 : Importer les Données et Appliquer ARIMA

Voici un exemple d'utilisation d'ARIMA pour modéliser les prix d'une action.

python

Copy code

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from statsmodels.tsa.arima\_model import ARIMA

# Simuler des données pour illustrer (utilisez vos propres données réelles ici)

np.random.seed(42)

dates = pd.date\_range(start="2023-01-01", periods=100, freq='B') # 100 jours ouvrés

prices = np.random.randn(100).cumsum() + 100 # Simuler une tendance pour les prix de clôture

price\_df = pd.DataFrame(data=prices, index=dates, columns=['Close'])

# Visualisation des prix de clôture

plt.figure(figsize=(10, 6))

plt.plot(price\_df['Close'], label='Prix de clôture')

plt.title('Prix de Clôture Simulés')

plt.show()

#### Étape 2 : Appliquer ARIMA pour la Prévision

Vous pouvez maintenant appliquer le modèle ARIMA à vos prix de clôture historiques pour faire des prévisions de prix futurs.

python

Copy code

# Paramètres d'ARIMA (p, d, q)

p = 5 # Nombre de retards dans l'auto-régression

d = 1 # Différenciation (pour rendre la série stationnaire)

q = 2 # Nombre d'erreurs dans la moyenne mobile

# Ajuster le modèle ARIMA

model = ARIMA(price\_df['Close'], order=(p, d, q))

model\_fit = model.fit(disp=0)

# Résumé du modèle ajusté

print(model\_fit.summary())

# Faire des prévisions pour les 30 prochains jours

forecast\_steps = 30

forecast, stderr, conf\_int = model\_fit.forecast(steps=forecast\_steps)

# Affichage des prévisions

forecast\_index = pd.date\_range(price\_df.index[-1] + pd.Timedelta(days=1), periods=forecast\_steps, freq='B')

forecast\_df = pd.DataFrame(data=forecast, index=forecast\_index, columns=['Forecast'])

# Visualisation des prévisions

plt.figure(figsize=(10, 6))

plt.plot(price\_df['Close'], label='Prix Historique')

plt.plot(forecast\_df['Forecast'], label='Prévisions ARIMA', color='red')

plt.fill\_between(forecast\_df.index, conf\_int[:, 0], conf\_int[:, 1], color='pink', alpha=0.3)

plt.title('Prévisions de Prix avec ARIMA')

plt.legend()

plt.show()

### Explication du Code

1. Modèle ARIMA :
   * Nous avons ajusté un modèle ARIMA avec les paramètres (p=5, d=1, q=2), qui signifie :
     + 5 retards dans la série (prix précédents utilisés),
     + 1 différenciation (pour rendre la série stationnaire),
     + 2 erreurs de moyenne mobile utilisées dans le modèle.
   * Vous pouvez ajuster ces paramètres pour obtenir le meilleur modèle pour vos données.
2. Prévisions :
   * Le modèle ARIMA est utilisé pour prévoir les 30 prochains jours de prix.
   * Les résultats sont affichés avec des bandes de confiance qui montrent l'incertitude des prévisions.
3. Graphique :
   * Le graphique montre à la fois les prix historiques et les prévisions futures du modèle ARIMA.

### Étape 3 : Identifier les Pics et Creux dans les Prévisions

Une fois les prévisions faites, vous pouvez identifier les points hauts (pics) et bas (creux) dans les prévisions à l'aide des techniques suivantes :

1. Identification des Pics et Creux :
   * Points Hauts (Pics) : Si une séquence de prévisions montre une forte augmentation suivie d'une baisse, vous pouvez considérer cela comme un pic.
   * Points Bas (Creux) : Si une séquence de prévisions montre une forte baisse suivie d'une augmentation, cela peut indiquer un creux.

#### Exemple de Détection des Pics et Creux dans les Prévisions

Vous pouvez utiliser une fonction simple pour détecter les pics et les creux dans les prévisions :

python

Copy code

from scipy.signal import argrelextrema

def find\_peaks\_and\_troughs(series, order=3):

"""

Trouve les pics et creux locaux dans une série temporelle.

Parameters:

series (pd.Series): Série temporelle

order (int): Nombre de points voisins à comparer

Returns:

dict: Indices des pics et des creux

"""

peaks = argrelextrema(series.values, np.greater, order=order)[0]

troughs = argrelextrema(series.values, np.less, order=order)[0]

return {"Peaks": peaks, "Troughs": troughs}

# Trouver les pics et creux dans les prévisions

peaks\_troughs = find\_peaks\_and\_troughs(forecast\_df['Forecast'], order=3)

# Affichage des pics et creux

print("Indices des pics : ", peaks\_troughs["Peaks"])

print("Indices des creux : ", peaks\_troughs["Troughs"])

# Visualiser les pics et creux sur le graphique des prévisions

plt.figure(figsize=(10, 6))

plt.plot(forecast\_df['Forecast'], label='Prévisions ARIMA', color='red')

plt.scatter(forecast\_df.index[peaks\_troughs["Peaks"]], forecast\_df['Forecast'].iloc[peaks\_troughs["Peaks"]], marker='^', color='green', label='Pics', s=100)

plt.scatter(forecast\_df.index[peaks\_troughs["Troughs"]], forecast\_df['Forecast'].iloc[peaks\_troughs["Troughs"]], marker='v', color='blue', label='Creux', s=100)

plt.title('Détection des Pics et Creux dans les Prévisions ARIMA')

plt.legend()

plt.show()

### Explication

1. Détection des Pics et Creux :
   * argrelextrema de scipy est utilisée pour identifier les pics (maxima locaux) et les creux (minima locaux) dans les prévisions de la série temporelle.
   * Les pics sont représentés par des triangles verts, tandis que les creux sont représentés par des triangles bleus.
2. Visualisation :
   * Le graphique montre les pics et creux identifiés dans les prévisions du modèle ARIMA. Vous pouvez utiliser cette information pour décider des moments potentiels d'achat ou de vente.

### Conclusion

En utilisant un modèle ARIMA, vous pouvez prédire les prix futurs d'une action et essayer d'identifier les moments où l'action pourrait atteindre son pic ou son plus bas au cours du mois. Couplé à des techniques de détection des extrêmes comme argrelextrema, cela peut vous aider à anticiper les mouvements de prix et à ajuster vos stratégies d'achat et de vente.