Projet 2: Trading algorithmique sur cryptomonnaies.

Dans le cadre du projet 2 de notre cours d'intelligence artificielle, nous avons été amenés à créer un bot de trading de cryptomonnaies.
Nous avons choisi de coder l'algorithme en python car il s'agit d'un des langages de programmation avec lequel nous avons l'habitude de travailler possédant de nombreuses librairies utiles nous permettant de créer facilement un bot fonctionnel.

Ce projet sera présenté en plusieurs parties. Tout d'abord, nous présenterons brièvement le déroulement du projet. Ensuite nous nous intéresserons à l'aspect stratégique du trading par algorithme, pour enfin finir par la partie technique du code et les élargissements possible.

Déroulement du projet :

La première étape de ce projet a été de faire des recherches sur le trading algorithmique, sur son fonctionnement, ses spécificités ou encore sur les bots de trading en général. Nous avons donc décidé de faire nos recherches séparément puis de mettre ensuite nos connaissances en commun afin de pouvoir ratisser le plus d'informations possibles. Après deux semaines de recherches, nous nous sommes donc réunis afin de commencer à débattre autour du bot et de commencer à avoir une idée assez précise de ce que nous voulions faire. Avec les recherches que nous avions faites, nous avions assez vite compris que pour créer notre bot, il nous fallait une partie du code qui servirait à se connecter à l'api d'un site de trading de cryptomonnaie (que ce soit pour récupérer des données en temps réel ou passer des ordres) et l'autre sur la stratégie de trading. Vu le temps imparti, nous avons décidé de nous concentrer surtout sur la partie connexion à l'API et de faire une stratégie ensuite en fonction du temps restant. Après avoir effectué un certain nombre de recherches que ce soit via des github, des vidéos ou encore des documentations de librairies existantes. Après ces recherches nous avons donc décidé d'utiliser l'API de Binance puisqu'elle était assez simple d'utilisation. Nous avons commencé par établir la connexion grâce à websocket afin de pouvoir récupérer toutes les données en temps réel puis de se connecter à Binance via la librairie python-binance. Enfin la dernière étape fut de choisir et adapter la stratégie à mettre en place, par soucis de temps nous avons opté pour une stratégie assez simple se basant sur l'indicateur Aroon. Pour ce qui est du travail d'équipe, à part pour les recherches de début de projet, nous avons décidé de travailler toujours ensemble via des réunions hebdomadaires afin d'avancer ensemble dans le projet et que l'un de nous ne perde pas le fil du projet si l'un avance de son côté.

Fonctionnement du bot:

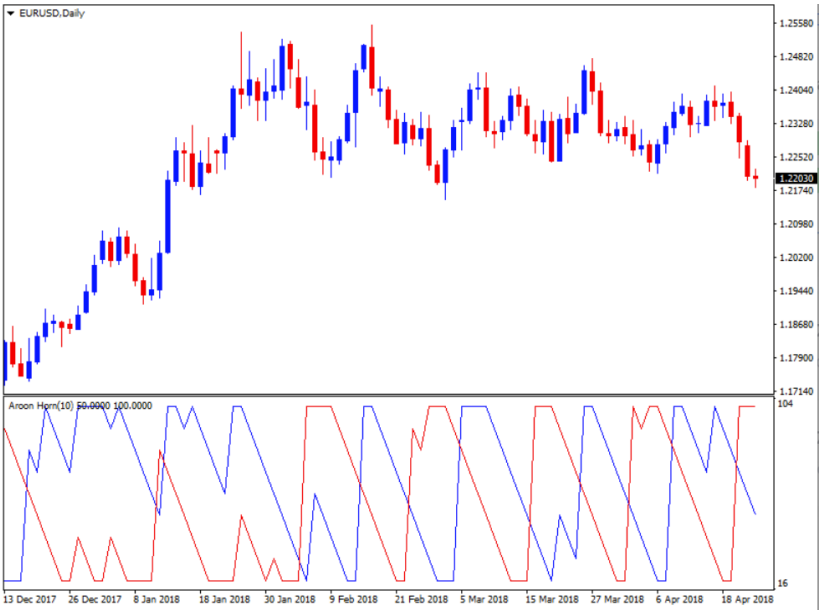
Le principe d'un algorithme de trading est une stratégie d'automatisation de vente et d'achat d'un sous-jacent par un programme informatique.
Le but de l'algorithme est de récupérer les valeurs de l'actif et de repérer les tendances du marché afin d'orienter notre choix d'achat ou de vente.

Afin de se connecter à Binance, nous utilisons l'API eponyme. Elle nous permet de nous connecter à notre compte, d'effectuer des ordres et de récupérer facilement des informations en temps réel. Cette API comporte plusieurs centaines de valeurs pouvant être récupérées qu'on l'on peut retrouver dans sa documentation.

Une fois connecté à la plateforme de trading, il nous faut établir une stratégie de trading.

Nous avons choisi d'uniquement considérer l'indicateur Aroon dans un premier temps.
Il se présente sous la forme d'un oscillateur borné entre 0 et 100 et composé de deux courbes :

- L'Aroon Up en bleu
- L'Aroon Down en rouge



Aroon Up = ((Période - Nombre de périodes depuis le plus haut) / Période) * 100
Aroon Down = ((Période - Nombre de périodes depuis le plus bas) / Période) * 1003

En ce qui concerne les deux courbes, nous pouvons les interpréter de la manière suivante:

Si la ligne Aroon Up est au-dessus de la ligne Aroon Down, cela démontre une forte tendance haussière. Si la ligne Aroon Down est au-dessus de la ligne Aroon Up, cela démontre une forte tendance baissière.
De plus, si la ligne Aroon Up est au-dessus de 70, cela montre une forte tendance haussière. Si la ligne Aroon Down est au-dessus de 70, cela montre une forte tendance baissière.

De même, si une ligne tombe sous le seuil des 50, cela signifie que la tendance en place s'affaiblit. Si une ligne passe en dessous de 30, cela annonce un possible retournement de tendance.

C'est le seul indicateur que nous avons utilisé car il est assez basique et il peut montrer des résultats facilement sur un algorithme de trading.

En revanche, pour perfectionner notre algorithme, il aurait fallu coupler cet indicateur avec d'autres données de marché pour avoir une stratégie plus juste et efficace.

Nous aurions voulu établir des règles d'achat et de vente en fonction du volume échangé, nous permettant de confirmer une tendance de marché: si le volume de trade augmente, les prix varient souvent dans la même direction.

SI nous avions eu plus de temps, nous aurions voulu utiliser les bandes de bollinger. Il s'agit d'un outil de mesure des oscillations qui fournit un grand nombre d'informations:

- Poursuite ou inversion de la tendance ;
- Les phases de consolidation du marché ;
- Les périodes de forte volatilité à venir ;
- Les sommets ou les creux de marché potentiels et les objectifs de prix.

Plus spécifiquement, cet indicateur est composé d'une bande supérieure, d'une bande inférieure et d'une ligne moyenne mobile. Les deux bandes latérales réagissent à l'action des prix du marché, s'élargissant lorsque la volatilité est élevée et se rapprochant lorsque la volatilité est faible.

Un autre indicateur que nous aurions pu aborder est le SMA (Simple Moving Average). Ce dernier fait la moyenne entre les dernières valeurs sur une période définie: si le SMA d'une période courte dépasse le SMA d'une période plus longue, cela signifie que l'on est sur une tendance haussière.

Partie technique :

```python
import websocket, json, pprint, talib, numpy
import config
from binance.client import Client
from binance.enums import *


SOCKET = "wss://stream.binance.com:9443/ws/btceur@kline_1m"

PERIOD = 15
Tendance_Forte = 70
TRADE_SYMBOL = 'BTCEUR'
TRADE_QUANTITY = 0.005

highli = []
lowli = []

dans_le_portefeuille = False

client = Client(config.API_KEY, config.API_SECRET, tld='us')
#client.API_URL = 'https://testnet.binance.vision/api' adresse pour passer sur le client test si voulu

def order(side, quantity, symbol,order_type=ORDER_TYPE_MARKET):
    try:
        print("sending order")
        order = client.create_order(symbol=symbol, side=side, type=order_type, quantity=quantity)
        print(order)
    except Exception as e:
        print("an exception occured - {}".format(e))
        return False

    return True
```

Tout d'abord on importe les librairies utiles pour le code, on va importer websocket afin de communiquer avec l'API binance, Talib pour les indicateurs afin de réaliser notre stratégie et binance afin de pouvoir exécuter des ordres sur notre compte binance.
L'URL de la variable socket permet d'avoir les données live de binance sur les crypto-monnaies souhaitées (ici nous avons choisi le BTC/EUR). Ensuite on peut aussi changer l'intervalle de temps voulu (ici chaque minute).
Ensuite on initialise les variables que l'on va utiliser par la suite :

- la Période correspond à la période que l'on va prendre pour utiliser notre indicateur (Aroon)
- La tendance forte est le nombre à atteindre qui montre que la tendance est forte (Pour l'indicateur aussi)
- le trade symbol qui correspond à la crypto monnaie que l'on va trade (Bitcoin contre euro pour notre code)
- Trade quantity représente la quantité que l'on va acheter ou vendre à chaque ordre
- Dans le portefeuille est la variable qui nous dit si on a la crypto-monnaie dans notre portefeuille ou non

Puis on initialise nos deux listes qui vont récupérer le High et le Low de la cryptomonnaie en temps réel dont on va se servir pour calculer l'indicateur.
Le client est ce qui va se permettre de se connecter à notre compte binance et donc d'y exécuter des ordres dessus.
La fonction Order est la fonction qui permet d'exécuter un ordre sur binance que ce soit un achat ou une vente (elle est donnée dans la documentation de python binance)

```
def on_open(ws):
    print('Connexion Ouverte')

def on_close(ws):
    print('Connexion Fermée')

def on_message(ws, message):
    try:
        global highli,lowli, dans_le_portefeuille
        print('Message Recu')
        json_message = json.loads(message)
        #pprint.pprint(json_message)
        candle = json_message['k']

        Low= float(candle['l'])
        High = float(candle['h'])
        highli += [High]
        lowli +=[Low]
        if len(highli) >  PERIOD:

            np_high = numpy.array(highli)
            np_low = numpy.array(lowli)
            aroondown, aroonup = talib.AROON(np_high,np_low, PERIOD)
            print("Tous les aroon calculé")
            print("aroondown :",aroondown)
            print("aroonup :",aroonup)
            last_aroondown = aroondown[-1]
            last_aroonup = aroonup[-1]
```

On a 3 fonction pour la connexion à la websocket :

- Le Open qui print un message quand on sera connecté à l'API
- le Close qui print un message quand on se sera déconnecté de l'API
- Le message qui sera exécuté à chaque message reçu via la Socket

C'est dans cette fonction message qu'on va trier les données reçues et établir notre stratégie de trading. On défini comme variable globale celle qu'on va modifier à l'intérieur de la fonction (les listes de low et high ainsi que la variable permettant de savoir si on a déjà la crypto-monnaie dans notre portefeuille) puis on charge le message via json. Ensuite on

récupère la colonne k du message qui correspond à toutes les données concernant les candles de notre crypto-monnaie. Puis on récupère seulement celles qui nous intéressent pour notre stratégie, c'est-à-dire le low et le high de chaque candle qu'on transforme en float (car l'indicateur doit recevoir des floats) et on ajoute chaque valeur à la liste correspondante. Ensuite, une fois qu'on a assez de données dans notre liste (ici on a choisi 15 car on jugé que cela était suffisant pour notre indicateur), on commence à calculer notre indicateur. Pour cela, on transforme nos listes en array (qui est le format souhaité pour l'indicateur) puis on utilise la librairie Talib afin de recevoir notre aroon up et aroon down qui vont permettre de mettre en place notre stratégie. Une fois calculé on prendra le dernier élément de chaque liste qui correspond au plus récent.

```
            if last_aroonup > last_aroondown and last_aroonup > Tendance_Forte :
                if not dans_le_portefeuille:
                    print("On achète car tendance haussière")
                    # ordre d'achat de la crypto
                    order_succeeded = order(SIDE_BUY, TRADE_QUANTITY, TRADE_SYMBOL)
                    if order_succeeded:
                        in_position = True
                else:
                    print("Déja acheté, on attend la vente avant de racheter")

            if last_aroondown > last_aroondown and last_aroondown > Tendance_Forte:
                if dans_le_portefeuille:
                    print("On vend car tendance à la baisse")
                    order_succeeded = order(SIDE_SELL, TRADE_QUANTITY, TRADE_SYMBOL)
                    if order_succeeded:
                        in_position = False
                else:
                    print("On ne peut pas vendre la crypto avant de l'avoir acheté")
    except Exception as e:
        print(e)

ws = websocket.WebSocketApp(SOCKET, on_open=on_open, on_close=on_close, on_message=on_message)
ws.run_forever()
```

Ici on a la partie stratégie où l'on utilise notre indicateur afin d'acheter ou de vendre notre crypto-monnaie. La stratégie est assez simple si l'indicateur aroon up est supérieur à celui aroon down (ce que signifie une tendance à la hausse) et qu'il est supérieur à 70 (ce qui signifie une tendance forte) alors c'est un signal d'achat et on achète la crypto-monnaie sauf si elle est déjà en notre position. Même stratégie pour la vente, si aroon down est est supérieur à aroon up (tendance à la baisse) et qu'il est supérieur à 70 (ce qui signifie une tendance forte) alors on vend la crypto-monnaie en notre possession sauf si nous n'en possédons pas.

Les dernières lignes servent à lancer la websocket afin d'envoyer les requêtes et récupérer les données sur Binance et de le faire tourner en continue afin de recevoir les messages indéfiniment.

Elargissement :

Si nous avions eu plus de temps, je pense qu'on l'aurait consacré à la stratégie du bot de trading. En effet il existe de très nombreux indicateurs permettant d'établir des stratégies simples comme très poussées. Nous avions choisi une stratégie de base afin d'avoir un bot fonctionnel le plus rapidement possible. Cependant, la stratégie est très simple et c'est pour cela qu'on aurait bien aimé parcourir tous les indicateurs possibles afin de créer notre propre stratégie. Mais c'est assez long à mettre en place puisqu'il faudrait réaliser de nombreux tests pour chacune des stratégies construites afin de voir leurs vraies efficacités. C'est pour cela que nous avions même pensé à l'utilisation du Machine Learning afin de potentiellement trouver quels étaient les indicateurs les plus performants ensemble. Même si ce n'est pas aussi simple que cela car il existe de nombreuses corrélations entre les indicateurs, il aurait donc probablement fallu faire une phase de tri ou élaborer un programme plus complexe afin de réaliser cette tâche (celle de trouver les indicateurs les plus performants).

Sources :

Binance Python API – A Step-by-Step Guide - AlgoTrading101 Blog
TA-Lib (mrjbq7.github.io)

hackingthemarkets/binance-tutorials: Real-Time Candlestick Charts and Crypto Trading Bot using Binance API and Websockets (github.com)

Python Examples of talib.AROON (programcreek.com)

Trader avec l'indicateur Aroon - aroon indicateur boursier - Admirals (admiralmarkets.com)

Binance Spot Test Network

https://binance-docs.github.io/apidocs

Documentation — Technical Analysis Library in Python 0.1.4 documentation (technical-analysis-library-in-python.readthedocs.io)
Aroon Indicator Definition and Uses (investopedia.com)

https://www.cointribune.com/analyses/strategie/les-meilleures-strategies-de-bots-de-trading-bitcoin-btc/

https://admiralmarkets.com/fr/formation/articles/indicateurs-forex/indicateur-rsi

https://www.investopedia.com/terms/s/sma.asp#:~:text=A%20simple%20moving%20average%20(SMA)%20is%20an%20arithmetic%20moving%20average,periods%20in%20the%20calculation%20average.&text=Short%2Dterm%20averages%20respond%20quickly,averages%20are%20slower%20to%20react.