CRYTPOCURRENCY TRADING BOT

INTRODUCTION

The cryptocurrency trading market is open 24x7, unlike our traditional equity markets, so many a time, it is observed that it's not always possible to scout for buying and selling opportunities, so I thought of creating a trading bot with the help of the **python** programming language, which would aid in increasing my knowledge about python and technical analysis skills of crypto-currency market.

ABOUT THE BOT

The bot is a simple Python code which operates indefinitely unless forced to shut down.

To begin, we must establish a connection between our own computer and the already existing data stream that flows from our crypto broker (Binance), to accomplish this task, I have used Python's **WebSockets** and for the technical analysis of our cryptocurrency, I have made use of the TA-Lib library, which provides users with a collection of Technical Analysis methods like the **RSI, ADX, MACD** and so on.

After a connection has been established between our computer and the broker, we need to dive into their data stream and collect relevant information which will help us to calculate our outcomes using our technical analysis indicators. In this project, I will be making use of the **RSI** indicator from TA-Lib to make decisions.

The data stream, the **Binance API ,** contains information about a lot of things but here in our project we will only be needed the information about the "Trade Symbol", i.e. the cryptocurrency we want to trade, the "Trade Quantity", i.e. how much quantity we want to trade and the "Closing Prices" of our candlesticks for the required crypto-coin.

TECHNICAL EXPLAINATION

A WebSocket object is made and configured with the address of the **URL** of the datastream from where it needs to constantly fetch data, the socket is made to run until its processes are forcefully shut down, the command of **.run_forever()**, allows us to achieve this state.

Now that we've established a connection and are retrieving data from the source in the form of a **JSON Object**, the data must be filtered and translated into appropriate data types upon which computations may be performed.

As previously said, we need to keep a track of our closing prices from our candlestick chart, therefore we make an **array**, to store our candlestick **closing prices**; the data in this array is the **primary info** that will assist us make purchasing and selling decisions.

Order placing is done by a function which connects the bot to our Binance account and then sends a message to the Binance Application with information about the **type of order** (buy/sell), **quantity** of the order to be executed, **symbol** or name of the crypto-coin and lastly the **order type**, i.e. limit orders or market orders.

We now create a conditional check, and this check will help us determine if the target crypto-coin is being overbought or oversold, and upon the complete execution of the check, our algorithm comes up with a solution as to whether buy or sell or do nothing with our current position and crypto-coin.

The program can run **indefinitely**, as it makes use of a function that makes the bot fetch data from the API stream as long as the target data-stream is online, longer the program runs, more accurate are the results produced.

RELATIVE STRENGTH INDEX (RSI) & PROJECT IMPLEMENTATION

The relative strength index (RSI) is a momentum indicator used in technical analysis that measures the magnitude of recent price changes to evaluate overbought or oversold conditions in the price of a stock or other asset.

In our project, we are making use of **RSI-14**, which takes **14 closing prices** as input and then using the mathematical formula, $RSI = 100 - [\frac{100}{1 + (Average\ gain)/\ (Average\ loss)}]$, the average gain or loss used in the calculation is the average percentage gain or loss during a look-back period. The formula uses a positive value for the average loss.

Upon the calculation of RSI, if our RSI value is **greater** than **70**, which is our threshold for a crypto-coin being **overbought**, we check if we are in position, and if the position flag is true, we sell our crypto-coin using the **order_succeded** function and after successfully selling our crypto-coin, we mark our position flag as false.

Similarly, if our RSI value is **less** than **30**, this indicates that the crypto-currency is **oversold**, here also we make a check if we are already in possession of this currency, and if **yes**, we do nothing, but if we do not own it, we make a call to the **order_succeded** function and put a buy up a **buy order** for the crypto-coin, and set our position flag as true.

CONCLUSION & RESULTS

The software was tested overnight with candlesticks closing at 1-minute intervals, and I was able to create a profit of **0.23$** by initial investing **$20.**

The project substantially aided me in gaining expertise in dealing with Python to construct bots, as well as knowledge of dealing with Web APIs and data collecting. The project also aided me in grasping several factors, which resulted in an increase in my knowledge about technical analysis.