ITP - Trading Bot 21/22

SUMMARY

1 Introduction - A short story

2 Manual for Telegram User

3 How to deploy live trading

**INTRODUCTION** - A short story

We had a certain way of trading…

Over time, we realized that we were using ***specific patterns*** to consider our trades. The goal was therefore to ***automate*** our trading.

For this we created **Bobi the trading bot**.

We can distinguish Bobi's creation into 3 parts:

* Bobi the Strategy Tester
* Bobi the trader live and 24/7
* Bobi the promoter on telegram

***1/ Bobi the strategy tester***

A strategy is deciding the right time to buy and then the time to sell a cryptocurrency to make a profit.

Before applying a strategy in real life, it may be wise to test it under past conditions. This is called doing a “backtest”.

So we set up Bobi to help us test these strategies with data from the past. For this, we used Python3, technical analysis libraries and we extracted existing data on FTX crypto exchangers.

Concretely, we allocated a fictitious amount of $1,000 to our program and simulated on past data, purchases when our purchase conditions were activated, and sales when our sale conditions were met.

Examples and explanations are below.

As a result of these simulations we obtain a lot of data such as:

The gain in %, the net gain in USD ($), the gain in % compared to the market, the highest drop in % of the portfolio, the best/worst trade in % and its closing date, …

This allowed us to refine our view of risk taking by testing our creations on many different cryptocurrencies in order to make our strategy more resilient.

**LT (long term) Strategy**

condition of ***purchase*** when:

* Exponential Moving Average 20 > Exponential Moving Average 400
* Super Trend == True (bullish/green)

then:

* the closing price of the previous candle > Exponential Moving Average 20
* the closing price of the current candle < Exponential Moving Average 20



condition of ***sale*** when:

* Exponential Moving Average 20 < Exponential Moving Average 400
* the closing price of the previous candle < Exponential Moving Average 20
* the closing price of the current candle > Exponential Moving Average 20



***2/ Bobi the live and 24/7 trader***

Now that we have improved and tested our strategy, we must implement it in real life and allow our program to trade live on the markets, night and day.

For this, FTX (an exchanger) allows us via its API, to connect and send requests to perform actions such as selling/buying.

It only took a few changes to our code to make it work. After several successful attempts, Bobi was finally ready.

***3/ Bobi the promoter on telegram***

Bobi is a successful trader, but a cumbersome character. He only speaks the Python language, so not everyone can chat with him.

This is why we allowed him to interact with users who did not want to bother with deploying and setting up the applications necessary to test Bobi's strategies.

For this, Telegram offers the creation of a bot via its API, allowing it to process messages from users joining Bobi's group.

They can test the different strategies on the crypto-currencies available there and see the results in a few minutes. Maybe the results will convince them…

**Manual for Telegram User**

/**help** :

receive help from Bobi

/**infos**:

Explains how to write a backtest instruction to the bot.

Basically Bobi needs informations in the following order:

* the strategy to use for the backtest (LT or ST)
* the currency to use, shortcuts like BTC, ETH, BNB, LTC, ATOM, AVAX, HOT, etc…
* the year to start with the backtest, written like 2017, 2019, 2020, 2022,...

/**example**:

gives you exemple on phone, you just need to click them.

/**about**:

gives you links to more documentations about Bobi’s creation.

**Manual to deploy live trading**

To deploy your trading bot you must follow the following steps:

Amazon Web Server

(https://aws.amazon.com/)

* create an account
* Click on ‘My management console account’
* search for ‘EC2’
* go to “instance”
* launch instances
* select an Ubuntu server (eligible for the free offer)
* launch the instance
* create a new key pair (for example: ‘mytestkey.pem’)
* download and store the .pem key on your computer.
* validate and launch the instance

Putty & PuttyGen

(<https://www.putty.org/>)

* download Putty & puttyGen
* open puttyGen and click on ‘Load’
* select and open your .pem key
* click on 'save private key' and save the generated private key (for example: 'mytestkey.ppk')

in Putty

* Copy/paste your Public IPV4 DNS address in 'Host Name', preceded by 'ubuntu@'

(eg: ‘ubuntu@ec2-12-34-56-78.eu-west-9.compute.amazonaws.com’)

* Click on SSH -> Auth
* Select your private key for authentication (.ppk)
* Click ‘Open’

Now, you are on your server.

On the server console, run the following list of commands:

*To update your software and install pip:*

$ sudo apt-get update

$ sudo apt install python3-pip

*To check the versions of your imports:*

$ python3 –version

$ pip –version

*To install the required libraries*

$ pip install pandas

$ pip install ta

$ pip install ftx

$ pip install ciso8601

*To create your python file:*

$ nano firstBot.py

Copy/paste the provided code in ‘firstBot.py’

FTX centralized exchanger

(<https://ftx.com/>)

* create an FTX account
* create a trading sub-account for the robot
* Parameter -> API
* retrieve private API keys

Then :

* return to firstBot.py with nano command firstBot.py
* Fill in the following fields with your own data:  
  **accountName**, FTXClient(**api\_key**, **api\_secret**, **subaccount\_name**)
* Also, choose the trading pair by changing the fields **pairSymbol** and **cryptoSymbol**
* save the code and exit the editor

Finally:

* run your code with the command:  
   python3 firstBot.py

Your Bot has made its first run.

To make it run continuously:

* Write the following command:  
  crontab -e
* Select default configuration
* Go under comments (#) and write the following code:

0 \* \* \* \* python3 firstBot.py

* save and exit the editor
* exit the console, your bot is running on the server.

**Enjoy !**