Binance-trade-bot

Binance Setup

- Create a Binance account (Includes my referral link, I'll be super grateful if you use it).
- Enable Two-factor Authentication.
- Create a new API key.
- Get a cryptocurrency. If its symbol is not in the default list, add it.

Tool Setup

Install Python dependencies

Run the following line in the terminal: `pip install -r requirements.txt`.

Create user configuration

Create a .cfg file named `user.cfg` based off `.user.cfg.example`, then add your API keys and current coin.
The configuration file consists of the following fields:

- **api_key** - Binance API key generated in the Binance account setup stage.
- **api_secret_key** - Binance secret key generated in the Binance account setup stage.
- **current_coin** - This is your starting coin of choice. This should be one of the coins from your supported coin list. If you want to start from your bridge currency, leave this field empty - the bot will select a random coin from your supported coin list and buy it.
- **bridge** - Your bridge currency of choice. Notice that different bridges will allow different sets of supported coins. For example, there may be a Binance particular-coin/USDT pair but no particular-coin/BUSD pair.
- **tld** - 'com' or 'us', depending on your region. Default is 'com'.
- **hourToKeepScoutHistory** - Controls how many hours of scouting values are kept in the database. After the amount of time specified has passed, the information will be deleted.
- **scout_sleep_time** - Controls how many seconds are waited between each scout.
- **scout_multiplier** - Controls the value by which the difference between the current state of coin ratios and previous state of ratios is multiplied. For bigger values, the bot will wait for bigger margins to arrive before making a trade.
- **strategy** - The trading strategy to use. See `binance_trade_bot/strategies` for more information

- **buy_timeout/sell_timeout** - Controls how many minutes to wait before cancelling a limit order (buy/sell) and returning to "scout" mode. 0 means that the order will never be cancelled prematurely.
- **scout_sleep_time** - Controls how many seconds bot should wait between analysis of current prices. Since the bot now operates on websockets this value should be set to something low (like 1), the reasons to set it above 1 are when you observe high CPU usage by bot or you got api errors about requests weight limit.

Environment Variables

All of the options provided in `user.cfg` can also be configured using environment variables.

```
CURRENT_COIN_SYMBOL:
SUPPORTED_COIN_LIST: "XLM TRX ICX EOS IOTA ONT QTUM ETC ADA XMR DASH NEO ATOM DOGE
VET BAT OMG BTT"
BRIDGE_SYMBOL: USDT
API_KEY: vmPUZE6mv9SD5VNHk4HlWFsOr6aKE2zvsw0MuIgwCIPy6utIco14y7Ju91duEh8A
API_SECRET_KEY: NhqPtmdSJYdKjVHjA7PZj4Mge3R5YNiP1e3UZjInClVN65XAbvqqM6A7H5fATj0j
SCOUT_MULTIPLIER: 5
SCOUT_SLEEP_TIME: 1
TLD: com
STRATEGY: default
BUY_TIMEOUT: 0
SELL_TIMEOUT: 0
```

Paying Fees with BNB

You can use BNB to pay for any fees on the Binance platform, which will reduce all fees by 25%. In order to support this benefit, the bot will always perform the following operations:

- Automatically detect that you have BNB fee payment enabled.
- Make sure that you have enough BNB in your account to pay the fee of the inspected trade.
- Take into consideration the discount when calculating the trade threshold.

Notifications with Apprise

Apprise allows the bot to send notifications to all of the most popular notification services available such as: Telegram, Discord, Slack, Amazon SNS, Gotify, etc.

To set this up you need to create a apprise.yml file in the config directory.

There is an example version of this file to get you started.

If you are interested in running a Telegram bot, more information can be found at Telegram's official documentation.

Run

```
python -m binance_trade_bot
```

Docker

The official image is available here and will update on every new change.

```
docker-compose up
```

If you only want to start the SQLite browser

```
docker-compose up -d sqlitebrowser
```

Backtesting

You can test the bot on historic data to see how it performs.

```
python backtest.py
```

Feel free to modify that file to test and compare different settings and time periods

Developing

To make sure your code is properly formatted before making a pull request, remember to install pre-commit:

```
pip install pre-commit
pre-commit install
```

The scouting algorithm is unlikely to be changed. If you'd like to contribute an alternative method,