ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO

BÁO CÁO TỔ CHỨC MÃ NGUỒN ỨNG DỤNG DỰ ĐOÁN XU HƯỚNG THỊ TRƯỜNG TIỀN MÃ HÓA SỬ DỤNG MÔ HÌNH HỌC MÁY.

Giảng viên: TS.Đặng Thanh Hải
Học phần: Chuyên đề công nghệ Nhật Bản
Mã LHP: INT3138 7

Nhóm sinh viên thực hiện:
Trần Mạnh Duy - 22026567
Phạm Khánh Linh - 21020080

Hà Nội, 2024

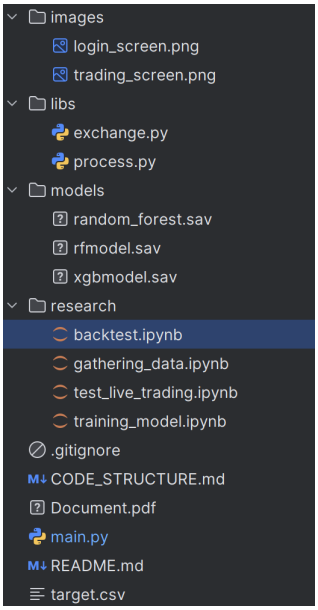
Mục lục

Danh sách hình vẽ

1 Giới thiệu

Mục tiêu báo cáo này là thể hiện cấu trúc mã nguồn của ứng dụng giao dịch tự động bằng cách sử dụng mô hình AI.

2 Tổng quan



Hình 1: Tổ chức mã nguồn ứng dụng.

Mã nguồn của ứng dụng được chia làm 3 thư mục:

1. Thư mục `libs`: chứa các đoạn mã lõi của ứng dụng, bao gồm các hàm thực hiện giao dịch, đăng nhập, lấy kết quả dự đoán từ mô hình AI.

2. Thư mục `models`: chứa các tệp mô hình đã được huấn luyện, nhằm mục đích thực hiện tính năng lựa chọn mô hình AI tùy theo người dùng để thực hiện giao dịch.

3. Thư mục `research`: chứa các tệp `ipynb` được sử dụng cho quá trình huấn luyện và kiểm tra, đánh giá hiệu quả của mô hình.

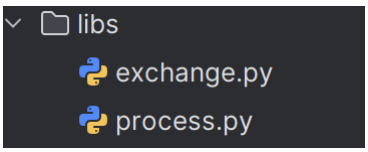
4. Ngoài ra còn có tệp `main.py` là đầu vào hoạt động của ứng dụng, tệp `README.md` và `CODE_STRUCTURE.md` cung cấp thông tin hướng dẫn sử dụng và tổ chức mã nguồn. Thư mục `images` cung cấp hình ảnh cho các tài liệu kể trên. Tệp `target.csv` chứa các dữ liệu đầu ra mong muốn được sử dụng để so sánh kết quả huấn luyện các mô hình AI.

3 Giao diện - Frontend

Mã nguồn giao diện được đặt trong tệp `main.py`, sử dụng thư viện `tkinter` là một thư viện giao diện trên `python`. Bao gồm:

- Lớp `Trade`:
  - thuộc tính `client` thể hiện tình trạng đăng nhập của người dùng vào ứng dụng.
  - thuộc tính `trading_info` thể hiện dữ liệu giao dịch của ứng dụng.
  - hàm `open_trading_window` với chức năng mở cửa sổ thông tin giao dịch. Từ cửa sổ này, người dùng có thể điền tên viết tắt của đồng tiền điện tử mà mình mong muốn giao dịch, bấm nút `Trade` để ứng dụng bắt đầu giao dịch.
  - hàm `trading` với đầu vào là ký tự thể hiện tên viết tắt của đồng tiền điện tử người dùng muốn giao dịch. Sau khi người dùng bấm nút `Trade`, ứng dụng sẽ gọi hàm này, hàm này sẽ gọi đến các hàm ở máy chủ - backend.
- hàm `login` sẽ thực hiện việc lấy thông tin private key, public key từ cửa sổ đăng nhập, sau đó tạo 1 đối tượng thuộc lớp `Trade` và gọi hàm mở cửa sổ thông tin giao dịch từ đối tượng đó.
- đoạn mã mở cửa sổ đăng nhập và duy trì hoạt động giao diện của ứng dụng.

4 Máy chủ - Backend



Hình 2: Mã nguồn máy chủ - backend.

Mã nguồn máy chủ được chia làm 2 tệp như sau:

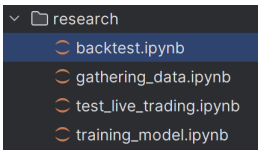
1. Tệp `exchange.py` thực hiện các thao tác liên quan đến giao dịch, bao gồm các hàm:
   - Hàm `check_login` nhận đầu vào là 2 chuỗi ký tự đại diện cho private key và public key của ví điện tử, hàm sẽ thực hiên kiểm tra tính hợp lệ của cặp khóa trên Client Testnet của Binance, nếu hợp lệ hàm sẽ trả về giá trị Client để thực hiện các giao dịch, nếu không hàm sẽ trả về False.
   - Hàm `balance_of` nhận đầu vào là Client của người dùng sau khi đăng nhập, cùng ký tự của đồng tiền điện tử, hàm sẽ trả về số lượng tiền điện tử ứng với ký hiệu đầu vào mà ví của người dùng đang sở hữu.

- Hàm `check_order` nhận đầu vào là Client người dùng đã đăng nhập, ký hiệu tiền mã hóa, và thao tác mua/bán, hàm sẽ trả về kết quả của lệnh giao dịch gần nhất của người dùng ứng với ký hiệu tiền mã hóa.
- Hàm `take_buy_order` nhận đầu vào là Client người dùng đã đăng nhập, kí hiệu tiền mã hóa, thao tác, và số lượng, hàm sẽ thực hiện giao dịch mua số lượng đầu vào ứng với ký hiệu tiền mã hóa.
- Hàm `take_sell_order` nhận đầu vào là Client người dùng đã đăng nhập, kí hiệu tiền mã hóa, thao tác, và số lượng, hàm sẽ thực hiện giao dịch bán số lượng đầu vào ứng với ký hiệu tiền mã hóa.

2. Tệp `process.py` bao gồm:

   - Đoạn mã thực hiện việc sử dụng mô hình.
   - Hàm `gather_data` nhận đầu vào là ký hiệu đồng tiền điện tử, thực hiện việc lấy dữ liệu tỉ giá đối với đồng USDT, khoảng dữ liệu cách 4 giờ trên thời gian thực. Sau đó hàm thực hiện chuyển dữ liệu thành các `data_frame`, tính toán các thuộc tính phục vụ cho việc đưa ra quyết định của mô hình đã chọn.
   - Đoạn mã nhằm sắp xếp các thuộc tính để đưa cho mô hình đưa ra kết quả.
   - Hàm `get_status` nhận dữ liệu đầu vào là `data_frame`, sau đó sử dụng các thuộc tính đã sắp sếp để đưa ra dự đoán, trạng thái trả về gồm 3 giá trị: 0 đại diện cho xu hướng không rõ, 1 đại diện cho xu hướng giảm, và giá trị còn lại đại diện cho xu hướng tăng.

5 Mã nguồn huấn luyện mô hình



Hình 3: Mã nguồn huấn luyện mô hình AI.

Mã nguồn phục vụ việc huấn luyện mô hình trong thư mục `research` bao gồm:

- Tệp `gathering_data.ipynb` thực hiện việc thu thập dữ liệu lấy trên mạng thử nghiệm của Binance, trong khoảng thời gian là từ ngày 1 tháng 6 năm 2020 đến ngày 1 tháng 11 năm 2022, khoảng dữ liệu cách nhau 4 giờ đồng hồ. Dữ liệu được lấy từ đồng tiền mã hóa Bitcoin. Dữ liệu này sau khi thu thập trở thành dữ liệu mẫu và được lưu trữ ở MongoDB ở localhost, để sử dụng đoạn mã này người dùng cần khởi động MongoDB tại máy tính của người dùng.

- Tệp `training_model.ipynb` sẽ thực hiện việc lấy dữ liệu từ MongoDB, bao gồm các dữ liệu Open - giá mở cửa trong một khoảng thời gian, Close - giá đóng cửa trong một khoảng thời gian, Low - giá thấp nhất trong một khoảng thời gian, High - giá cao nhất trong một khoảng thời gian, Volume - khối lượng giao dịch trong một khoảng thời gian và Time - khoảng thời gian. Ở các dòng mã tiếp theo sẽ thực hiện việc tính toán các thuộc tính và các chỉ số trên dữ liệu mẫu, làm trơn dữ liệu và thực hiện huấn luyện. Cuối cùng sẽ thực hiện so sánh đối với đầu ra huấn luyện với dữ liệu mục tiêu trong tệp `target.csv` để kiểm tra hiệu suất.
- Tệp `test_live_trading.ipynb` sẽ thực hiện kiểm tra khả năng giao dịch của mô hình đã huấn luyện trên dữ liệu thời gian thực.
- Tệp `backtest.ipynb` sẽ thực hiện kiểm tra khả năng giao dịch của mô hình đã huấn luyện trên khoảng thời gian khác so với dữ liệu mẫu.