Alpaca Trading Systems (Group 56)

Instructions

1. Clone the repository
2. Create a virtual env
3. Install all packages from requirements.txt
4. create a config directory in the repository
5. Setup the configuration as instructed below
6. Run main.py

Configuration

To set up the Alpaca API, create a `config.yaml` file in the `config/` directory with the following structure:

```
alpaca:
  data_api: <data_api>          # URL for Alpaca's data API
  paper_api: <paper_api>        # URL for Alpaca's paper trading API
  api_key: <your_api_key_here>  # Your Alpaca API key
  secret_key: <your_secret_key_here>  # Your Alpaca secret key
```

data_api: https://data.alpaca.markets/v2 (https://data.alpaca.markets/v2) paper_api: https://paper-api.alpaca.markets (https://paper-api.alpaca.markets)

Project Structure

Module/File	**Description**
api	Make calls to external Alpaca API
config	You need to make a config/config.yaml file which contains the api configuration keys
data	contains downloaded data for backtesting and offline use
tests	test for various helper functions
utils	various helper functions
config_loader	use this file to load the configs from the .yaml file
jobs	implementations of intra-day trading strategies

Strategy

This code implements a cross-sectional momentum trading strategy for cryptocurrency trading using the Alpaca API. Here's a brief explanation of the strategy and the workflow: (crypto_cross_trading_v2.py)

Strategy Overview

```
1) Cross-Sectional Momentum:
    The strategy ranks a set of cryptocurrencies based on their momentum over a specific period (6 hours).
    It identifies the asset with the highest momentum and trades it.

2) Momentum Calculation:
    Returns: Calculate the short-term percentage changes for each symbol.
    Momentum: Compare the return of the most recent period to a past period (1-hour return over 7 hours ago),
    and rank the cryptocurrencies.
    The highest-ranked asset is chosen as the "Buy" candidate.

3) Trading Logic:
    If no positions exist, buy the cryptocurrency with the highest momentum.
    If there's a position in a previously chosen cryptocurrency but none in the current top asset,
    sell the old one and buy the new top asset.
```

Workflow

```
1) Data Gathering:
    Retrieve historical price data for selected cryptocurrency pairs over the past 6 hours.
    Calculate close-to-close returns and momentum indicators.

2) Ranking Assets:
    Rank cryptocurrencies based on calculated momentum to determine the top-performing asset.

3) Trading Decisions:
    Identify the cryptocurrency with the highest momentum (buy_symbol).
    Compare current positions:
        If there's no position, initiate a buy order for the top-performing asset.
        If holding a position in a previously selected asset, close it before opening a position in the new one.

4) Order Execution:
    Use Alpaca's API to check account balances and positions.
    Submit market orders for buying and selling based on the strategy's decision.

5) Iteration:
    Run the trading logic every 5 seconds for 60 minutes.
```

Testing

We created another alpaca paper trading account to test our strategies and project.

1. Deployed the strategy in a simulated live environment.
2. Ran the strategy over a specific period: 10 minutes, 1 hour, 3 hours
3. Analyzed the results.

Testing data is stored in **data/testing_trade_data.txt**