BTC/USD Trading Strategy Backtesting Challenge

Objective

Design, implement, and rigorously backtest a **trading strategy** for Bitcoin (BTC/USD) while ensuring **no lookahead bias**.

Key Challenge Components

1. **Develop a Trading Strategy** – Create a robust strategy.
2. **Implement in Python** – Use the provided backtesting framework.
3. **Ensure No Lookahead Bias** – Critical for valid backtesting.
4. **Analyze Performance** – Evaluate using key metrics (Sharpe Ratio, Net Profit, Win Rate, Max Drawdown, etc.).

Key Details

Parameter	Value
Initial Capital	$1,000 (fully deployed per trade)
Benchmark	Buy-and-Hold (Long at start, exit at end)
Brokerage Fee	0.15% per trade (entry & exit)

Note: The provided BTC_2019_2023_1d.csv data is for **training only**. Your strategy will be evaluated and you will be ranked on **unseen test data**—avoid overfitting to the training set.

Trade Execution Logic:

At each time step i, the strategy:

- Processes data up to and including candle i
- Generates a signal

- Executes the trade at the **open of candle i+1**

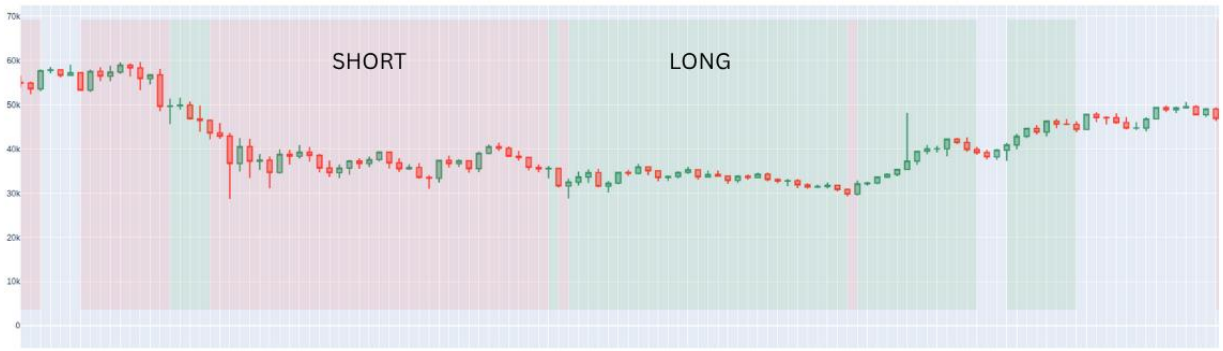
Provided Files

- backtester.py (DO NOT MODIFY)
- BTC_2019_2023_1d.csv (Historical OHLCV data)
- main.py (Modify process_data() and strat())

Note: A simple example strategy is included in main.py to help you get started. It is intentionally kept basic, and you are expected to develop more effective and higher-performing strategies.

Graph Interpretation

Running your strategy generates a **Trade graph and a PnL graph.**



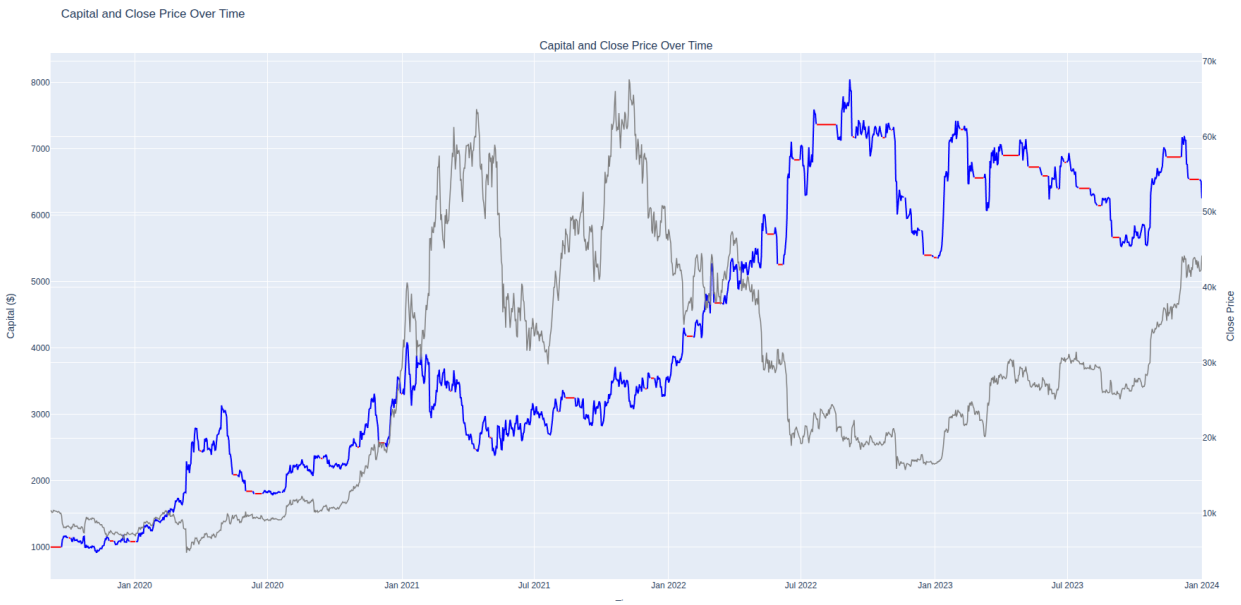
Trade graph

What it shows:

- **BTC/USD candlestick price chart**
- **Shaded background: Positioning**
  - 🟩 **Green background** = periods your strategy is in a **long position**
  - 🟥 **Red background** = periods your strategy is in a **short position**

This visual helps you quickly evaluate:

- How well your strategy rides trends
- Whether it's entering/exiting positions at sensible times



PnL graph

What it shows:

This chart visualizes the performance of your trading strategy against the price of Bitcoin over time.

🔹 *Plot Elements:*

- **Blue Line**: Your portfolio's **capital ($)** over time
- **Gray Line**: The **BTC/USD close price** (on a secondary y-axis)
- **Red line**: No open trades (current position =0)

📌 Interpretation:

- The **blue equity curve** reflects how your capital grows or shrinks with your strategy's trades.
- The **gray price curve** shows BTC's price movement to help visually compare your strategy to a buy-and-hold benchmark.
- Each **red marker** indicates where your strategy doesn't take any trades.

Your Tasks

1 process_data(data) Implementation

- **Purpose:** Compute all necessary technical indicators (e.g., SMA, RSI, MACD).
- **Requirement:** Use **only past & current data** (up to index i) to prevent future data leakage.

2 strat(data) Implementation

- **Purpose:** Generate trading signals in a new **"signals"** column using your strategy.
- **Signal Definitions:**

Signal	Action	Position After
`1`	**Go LONG** if currently neutral (0), or **Exit SHORT** if currently short (-1)	`+1` or `0`
`-1`	**Go SHORT** if currently neutral (0), or **Exit LONG** if currently long (+1)	`-1` or `0`
`2`	**Reverse from SHORT to LONG**	`+1`
`-2`	**Reverse from LONG to SHORT**	`-1`
`0`	**HOLD** your current position — no change	Same as before

Requirements:

✓*Track current position state.*

✓*Define clear entry/exit conditions.*

✓*(Optional) Include risk management (e.g., trailing stop-loss).*

❌ ***NO lookahead bias*** *– Signal at i must use only data up to i.*

Backtest & Analyze

Running main.py will generate:

1. final_data.csv

2. **Performance Report** (Sharpe Ratio, Win Rate, Max Drawdown, etc.)
3. **Lookahead Bias Check** (Pass/Fail)

Submission Guidelines

Modify Only:

- process_data() and strat() in main.py.

Submit:

1. Your final main.py file(s).
2. A **short PDF report** covering:
    a. Strategy logic (Write a short paragraph on your hypothesis)
    b. Key performance metrics
    c. Comparison/ranking if submitting multiple strategies.

◇ Multiple Strategies?

Allowed! Submit as main1.py, main2.py, etc., and **order by Sharpe Ratio** in your report.

Deadline & Grading - Deadline: 11:59 PM, 27th July

Grading Criteria:

✓ Strategy logic
✓ Performance metrics
✓ Clean implementation
✓ Confirmed absence of lookahead bias

Getting Started

1. Download the provided files.
2. Implement your strategy in main.py.
3. Run and test frequently.
4. Submit main.py + report by the deadline.

Good luck — let your trading ideas shine!