Group Project

Algorithmic Trading with Reinforcement Learning

OEOD, Master in Data Science, Iscte - IUL

Diana A. Mendes

November 14, 2024

Deadline: 29 December, 2024

Oral presentation : 15 January, 2025

---

1 Project Goals

- Define **2 trading strategies**: one based on **statistical/machine learning models** and the second one based on **reinforcement learning algorithms** for the historical stock prices attributed to your group.
- Use at least **two evaluation metrics (backtesting)**, and compare and interpret the obtained results.

1.1 Introduction

- Algorithmic trading (AT) refers to the process by which traders consistently buy and sell one given financial asset to make profit. It is widely applied in trading stocks, commodity futures and foreign exchanges.
- *Trade-off between highest return and lowest risk*: If an investor is expecting to invest in a riskier investment option than the risk-free rate then he/she is expecting to earn more in return.

- This is to compensate for the risk that the investor is taking.
- There exists a *risk-free rate* which is the rate that an investor earns on his/her investment without taking any risk.
- Refer to these transactions as *assets*.

1.2 What to do?

- Obtain/Extract the historical stock prices of the assets.
- Calculate the returns, expected mean returns, and the risk.
- Define statistical trading strategy (at least one, maximum 2).
- Evaluate the efficacy of the trading strategy.
- Use at least one reinforcement learning algorithm for trading the assets (Q-learning, A2C, MActor).
- Evaluate the efficacy of the trading strategy.
- Fine-tune your algorithms.
- Compare and interpret the results.

1.3 Report and Scripts

- Write a report (less than 25 pages) where you should have: an introduction, the project problem, methodology, data, results and conclusions.
- Please do not colocate Python outputs in the report (except for figures).
- All secondary information can be written in the Appendix.
- The Python Notebook ou Script without errors, with explicative comments, and able to run as-is.
- The report (in **pdf** format) and the Python file should be submitted to Moodle before the deadline (29 December, 2024)

1.4 Python Libraries

- pandas
- numpy
- matplotlib
- yfinance - download historical price data

- talib, quantstats - technical indicators and analysis
- mplfinance - matplotlib utilities for financial data visualization
- backtesting
- Gymnasium, gym, gym-anytrading

2 Data

Please analyze data from the last 5/6 years (from 2019-01-01 to 2024-10-31)

- Examples of assets, exchange rates and commodities that can be analysed and traded.

Tickers 'TSLA', 'AAPL', 'KO', 'FB', 'WMT', 'BTC-USD' , 'HSY', 'MSFT', 'GOOG', 'IBM', 'AXP', 'BA', 'MRK', 'CL=F', 'GC=F', 'NG=F', 'NVDA', 'NVDA','XOM', 'DIS'

3 Some definitions

Asset Returns/Rate of Returns. The cumulative daily rate of return is useful to determine the value of an investment at regular intervals.

$$\mathbf{r}_t = \text{ Return For Each Day } = \log\left(\frac{\text{Today's Price}}{\text{Yesterday's Price}}\right)$$

```
# compute returns for price series in data-frame 'data'
# use function 'pct_change' from pandas
import numpy as np
import pandas as pd
returns = data.pct_change()
## ou
returns=np.log(data).diff().dropna()
```

Asset Expected Mean Returns

$$\text{Asset Expected Return } = \frac{\text{Sum (Returns)}}{\text{Total Number Of Observations}}$$

Volatility

- Market volatility is defined as the standard deviation of prices/returns.

$$VOL = \sigma_p[\mathbf{r}]$$

- Market volatility increases the risk of the investment.
- A risky asset is an asset that provides uncertain monetary flow to its owner.

- Trading strategies involving shorter trading time frames work more effectively in volatile markets.

Gain-Loss Ratio - GLR

GLR represents the relative relationship of trades with a positive return and trades with a negative return **r**

$$GLR = \frac{\mathbb{E}[\mathbf{r} \mid \mathbf{r} > 0]}{\mathbb{E}[-\mathbf{r} \mid \mathbf{r} < 0]}$$

Sharpe Ratio - SR

- Sharpe Ratio is the amount of *excess return* (the return earned by stock and the risk-free rate, which is usually estimated using the most recent short-term government treasury bill) over the risk-free rate as the relevant measure of risk.

$$\text{Sharpe Ratio } = \frac{\text{Portfolio Expected Return } - \text{ Risk Free Rate}}{\text{Portfolio Volatility (Standard Deviation)}}$$

- A large Sharpe Ratio indicates that the stock's excess returns are large relative to the stock's volatility.
- The greater the Sharpe Ratio, the better its risk-adjusted performance.
- Usually, an SR greater than 1 is acceptable for the investors, 2 is very good and 3 is excellent.
- A negative Sharpe ratio means the risk-free or benchmark rate is greater than the historical return, or else the return is expected to be negative.
- The annualized Sharpe Ratio is calculated using the following formula:

$$SR = \sqrt{252}\frac{R_p - R_f}{\sigma_p}$$

- where R_p is the average daily return, R_f is the risk-free rate of return (ignored here as it is very small) σ_p is the standard deviation of the return (i.e., the volatility or the risk)

```
# daily risk free rate
rf = (1.02**(1/360))-1
# Calculate volatilities, expected returns and sharpe ratios.
volatility = returns.std() # volatility
exp_returns = returns.mean() # excpected returns
sr = (exp_returns-rf)/volatility # Sharpe ratio
```

CAGR

- Compound Annual Growth Rate (CAGR), provides a constant rate of return over the time period.

- So, CAGR tells us what we have at the end of the investment period.
- CAGR is calculated by the following formula:

$$(EV/BV)^{1/n} - 1$$

- where EV - ending value, BV - beginning value and n - number of years.
- CAGR does not reflect investment risk.
- It is essentially a number that describes the rate at which an investment would have grown if it had grown at the same rate every year and the profits were reinvested at the end of each year

```
# define function that compute CAGR
def cagr(start_value, end_value, num_periods):
    return (end_value / start_value) ** (1 / (num_periods - 1)) - 1

# example
start_value = float(df.iloc[0])
end_value = float(df.iloc[-1])
num_periods = len(df)
result = cagr(start_value, end_value, num_periods)
print(result)
```

Moving Average - MA ou Simple Moving Average - SMA

- A q-day moving average is, for a price series x_t and a point in time t, the average of the past q days: that is, if $MA(q)$ denotes a moving average process, then:

$$MA(q) = \frac{1}{q} \sum_{i=0}^{q-1} x_{t-i}$$

- Moving averages smooth a series and help identify trends.
- The larger is, the less responsive a moving average process is to short-term fluctuations in the series.
- The idea is that moving average processes help identify trends from "noise".

3.1 Trading strategies

- Fundamental analysis is the study of external factors that influence the price of an asset, such as economic developments, political decision-making, market sentiment, and other events.
- Technical analysis is the study of past prices to predict future prices. Technical analysts use charts and technical indicators to interpret price action.

- Put differently, technical analysis tries to answer the question of when to buy/sell an asset.
- The three most important basic principles of technical analysis are:
- The market ultimately discounts everything: thus, all data is contained in the price.
- Prices move either in a trend (up/down) or sideways.
- History repeats itself; this is the reason for support and resistance zones and patterns in charts.

Simple moving averages strategy

$$\text{SMA}(N)_t = \frac{P_{t-N+1} + P_{t-N+2} + P_{t-N+3} + P_t}{N} = \frac{1}{N}\sum_{i=1}^{N} P_{t-N+i}$$

where P_t is the stock price at time moment t and N is the number of days to consider for smoothing.

- For example, use two moving averages (MA), one considered to be fast, and the other slow.
- The strategy is: Trade the asset when the fast MA crosses over the slow MA.
- Exit the trade when the fast MA crosses over the slow MA again.
- A trade will be prompted when the fast MA crosses from below to above the slow MA, and the trade will be exited when the fast MA crosses below the slow MA later.
- If the fast MA is above the slow MA, this is a bullish regime (the bulls rule), and a bearish regime (the bears' rule) holds when the fast MA is below the slow moving average

Exponentially weighted moving averages strategy

$$\text{EMA}(N)_t = \alpha P_t + (1-\alpha)\,\text{EMA}(N)_{t-1}$$
$$\alpha = \frac{2}{N+1}$$

```
# simple moving average - use rolling function
short_rolling = data.rolling(window=20).mean()
# exponential moving average - use ewm function
short_exp_rolling = data.ewm(span=10, adjust=False).mean()

# example
# EMA for 10 and 20-day windows
df['ema_short'] = df['Close'].ewm(span=10, adjust=False).mean()
df['ema_long'] = df['Close'].ewm(span=20, adjust=False).mean()
# define a crossover strategy
df['bullish'] = 0.0
df['bullish'] = np.where(df['ema_short'] > df['ema_long'], 1.0, 0.0)
df['crossover'] = df['bullish'].diff()
```

RSI strategy

Using the Relative Strength Index, you can identify whether a price trend is overbought or oversold. It measures the speed and change of price fluctuation on a scale of 0 to 100, providing insights into overbought or oversold conditions, as well as potential trend reversals. (80-20 RSI Trading Strategy)

Relative Strength Index (RSI) can be calculated by the following expression:

$$RSI = 100 - \left[\frac{100}{1 + \frac{\text{Average gain}}{\text{Average loss}}}\right]$$

where the average gain or loss used in this calculation is the average percentage gain or loss during a look-back period. The standard number of periods used to calculate the initial RSI value is 14.

```
def get_rsi(returns, lookback):
    up = []
    down = []
    for i in range(len(returns)):
        if returns[i] < 0:
            up.append(0)
            down.append(returns[i])
        else:
            up.append(returns[i])
            down.append(0)
    up_series = pd.Series(up)
    down_series = pd.Series(down).abs()
    up_ewm = up_series.ewm(com = lookback - 1, adjust = False).mean()
    down_ewm = down_series.ewm(com = lookback - 1, adjust = False).mean()
    rs = up_ewm/down_ewm
    rsi = 100 - (100 / (1 + rs))
    rsi_df = pd.DataFrame(rsi).rename(columns = {0:'rsi'}).set_index(
                                            returns.index)
    rsi_df = rsi_df.dropna()
    return rsi_df[3:]
```

On Balance Volume - OBV

On Balance Volume, which is a technical trading momentum indicator that uses volume flow to predict changes in stock price

$$OBV(t) = OBV(t-1) + \begin{cases} \text{Volume } (t) & \text{if } P_t > P_{t-1} \\ 0 & \text{if } P_t = P_{t-1} \\ -\text{Volume } (t) & \text{if } P_t < P_{t-1} \end{cases}$$

where $OBV(t)$ - current on-balance volume level, $OBV(t-1)$ - previous on-balance volume level and Volume - latest trading volume amount.

Other trading strategies

- MACD (Moving Average Convergence Divergence) crossover strategy

- Williams R% strategy
- Stochastic Oscillator
- Bollinger band strategy
- Pairs trading strategy
- Machine Learning prediction-based strategies

Other notes

- Indicators can be used to define strategies ou as explanatory variables (for predicting stock prices)
- Backtesting - how profitable the strategy is on historical data (evaluate the quality of the strategy)
- Benchmark the strategy, or compare it to other available (usually well-known) strategies in order to determine how well we have done.
- Cumulative returns on our strategy give the evolution of the portfolio worth over the time-period of running the strategy.
- The flat periods represent periods when we held no assets.
- Daily Value-at-Risk, it is a popular risk metric. It indicates that in 95% of the cases, we will not lose more than 0.5% by keeping the position/portfolio for 1 more day.

4 Bibliography

[1]. Yves Hilpisch, (2021), Python for Algorithmic Trading, O'Reilly Media, Inc.