CFA FNQ PS

By Divyansh Gupta

PS DESCRIPTION

Our task was to design, implement, backtest and evaluate quantitative trading strategy on an Indian equity, using OHLCV data. I decided to work on a trend following strategy (Track 2 out of given tracks). The basic hypothesis behind this strategy was to believe that there are periods of trend/momentum in market, identify those trends and trade accordingly.

Downloading and Cleaning the Data

To begin, I needed to choose stocks that were suitable for the trend following strategy. I started by considering 10 good stocks from various sectors (like SBI, RELIANCE, TATAMOTORS) and downloaded their OHLCV data of 5 years (from 2020-2025) using yfinance library. I saved that data as csv to prevent future API calls for using the same data. Then, although yfinance gives fairly clean data, before using the data, it needed to be cleaned. First of all, the data consisted of MultiIndex columns, so I converted them to Index columns, with names becoming tuples. Then I dropped all NaN values.

Inspecting and Visualizing the data

Now, I used describe function to see various info about the data at hand. Then I also defined one helper function to plot a given datafield and used it to plot close values. But they were absolute, so not much could be inferred. Then I defined a helper functions to normalize datafields and used it to normalize close and volume data and plotted it. From both these analysis I deduced that TATAMOTORS is a good stock, because it had high (std dev)/(mean) ratio, which basically means that it is more volatile, it’s price fluctuates more as compared to others, giving rise to more and better trading opportunities, in this case, trends/momentum. Also it had fairly high volume traded.

Extracting Data of desired stock and creating helper functions

After deducing a suitable stock for our strategy, I defined a helper function to extract data of a single stock out of the overall data. I used that to extract all the data related to TATAMOTORS.NS and stored that in variables. I also defined helper functions to help implement our trading strategy. After all this, it was time to finally write down our strategy. I used backtesting.py library to backtest the strategy over historical data, so I needed to implement my strategy as an inherited class of Strategy class offered by the library, with a constructor and a next() method.

Exploring the Indicators

I had a basic idea of trend following (momentum) strategies. I started by reading about the suggested technical indicators - EMA/MACD and ADX. I explored that when short term EMA (9 Days) crosses over longer one (21 Days), it’s considered a buy signal and vice versa. Similarly, when MACD line (9 Days) crosses MACD signal line (difference between 12 and 26 days), it indicates a bullish market. Further ADX, along with +DI and -DI confirms that a market is in trend and whether it is bullish or bearish. I further used ATR and SMA of volume for trend confirmation and exit condition.

Defining the Strategy

I then tried to create strategy out of these indicators. When MACD line crosses over MACD signal line, short EMA crosses over long EMA, ADX is above threshold (25), +DI is greater than -DI and Volume is at 1% surge as compared to previous 5 day SMA, then it will be considered as long signal. Then if the conditions are opposite, then it's considered short signal. For exit conditions, if the ADX decreases below a certain level (20), or price decreases (or increases in case of short position) below a certain point (determined by subtracting a multiple of ATR), position is closed. Further take-profit and stop loss are applied, based on ATR to secure profit and minimize loss.

Long - Short Mechanism

Finally, I implemented Buy-Sell Mechanism. There can be three positions - long, short or none. If the position is long then - if the signal suggests buying, no change in position, if the exit condition is triggered, position is closed, and if selling is suggested, position is closed then, short position is acquired. Similarly, if the position is short then - if the signal suggests buying, position is closed then, long position is acquired, if the exit condition is triggered, position is closed, and if selling is suggested, no change in position. If the position is none, then trading is done based on entry conditions' suggestion - if it suggests long, long position is acquired and vice versa.

Backtesting

I used backtesting.py library to backtest the strategy on historical data of TATAMOTORS.NS from 1st Jan 2020 to 2025. I employed a commission of 0.2%. The results were stored in a variable, which I saved as csv for future reference. The metrics included in results were returns, drawdown, sharpe ratio, sortino ratio, win rate, and so on. Similarly, I saved the equity curve as png. Now I needed to introduce changes based on results continuosly until I get satisfactory results.

Evaluation of results and improvemnets

When I initially tested the strategy, 0 trades were performed in a total of 5 years! I then came to the conclusion that the conditions I implemented were too restrictive. I then started to experiment by removing various conditions. I came to conclusion that I should use either of EMA or MACD. Adding OR instead of AND seemed to do better by doing 15 trades and 0.49 sharpe ratio. Removing volume or DI conditions increased the number of trades but subsequently decreased sharpe ratio, so I considered keeping them. Then, I experimented with values of ADX threshold and ATR multiples for stop-loss and take-profits to get a sharpe of 0.577, with a win rate of 60%. Finally, I tweaked period of short and long EMA to get 15 trades, 57.89% win rate and 0.663 sharpe.

References

The References used are as follows:

1. chatgpt.com
2. perplexity.ai
3. finance.yahoo.com
4. google.com
5. investopedia.com
6. https://www.youtube.com/watch?v=GDMkkmkJigw