Building & backtesting a stock trade algorithm with Python

Introduction

I chose this project because I am interested in how technology and finance intersect. Algorithmic trading enables programming skills to make data-driven decisions in the stock market. This allowed me to combine Python programming with my interest in finance to explore real-world trading strategies.

Background

- ❖ Algorithmic trading: Using computer programs to automatically execute trades based on its predefined rules
- ❖ Risk vs Return: Measures how much potential loss is taken for a certain level of return.
- ❖ Why Python: It has powerful libraries like pandas, numpy, yfinance, and matplotlib for data analysis, visualization, and backtesting strategies.

Methodology

Step 1 - Collected Data

- I collected daily stock prices for Google, Apple, and Tesla using Python and the Yahoo Finance API. The data covers the period from 2018 to 2025.

Step 2 - Indicators:

- I calculated the 20-day and 50-day moving averages to identify short-term vs long-term.

Step 3 - trading strategy:

- My strategy was simple: buy when MA20 > MA50, stay in cash otherwise.

Step 4 - Backtesting:

- I simulated trades with an initial cash of $10,000 and calculated the daily portfolio value using Python.

Step 5 - Performance Metrics:

- Using my code, I calculated cumulative returns, annualized the return, annualized volatility, the Sharpe ratio, and maximum drawdown.

Step 6 - Visualization

- I created plots of portfolio value for each stock to visualize how the strategy performed over time.

Results

Stock	Cumulative Return	Annual Return	Volatility	Sharpe	Max Drawdown
GOOGL	141.94%	14.27%	21.92%	0.65	-38.12%
AAPL	201.39%	17.05%	21.19%	0.80	-29.09%
TSLA	423.41%	33.35%	47.20%	0.71	-60.63%

Analysis

Google

- The Sharpe ratio of 0.65 indicates the strategy produced decent returns relative to risk.
- The max drawdown of -38% shows the largest loss during the period.

Apple

- The Sharpe ratio of 0.80 indicates efficient risk-adjusted performance.
- Max drawdown of -29% shows relatively lower risk compared to Google and Tesla

Tesla

- High volatility has led to larger fluctuations in the portfolio.
- Max drawdown of -60% reflects the riskier nature, but cumulative return is the highest, which is 423%.

Reflection

Through this project, I learned:

- With the help of Google Gemini in Colab, to explore new coding concepts and how to debug code.
- I strengthened my Python skills with pandas, numpy, matplotlib, and rolling windows.
- I learned backtesting basics and how to simulate portfolio growth.
- I started understanding risk metrics like the Sharpe ratio and max drawdown.
- This project gives me an insight into how finance and computer science intersect in quantitative trading.

Next steps

- Right now, I've only tested Google, Apple, and Tesla. Next, I could try adding more stocks or ETFs to see if my strategy works across different markets.

- Currently, I've only used MA20 and MA50 to generate buy and sell signals. I could add other indicators to improve my strategy and get more accurate signals, which shows a deeper understanding of trading strategies.
- The simulation only assumes zero trading costs since, in real trading, buying and selling stocks have fees or price impacts like slippage. Including transaction costs and sillage would make my backsets appear more realistic.
- So far, I've simulated trading in each stock individually, starting with $10,000 per stock. I could stimulate a diversified portfolio by splitting cash across several stocks. Which would show my understanding of risk management and diversification.