Python in Finance
Quantitative Trading Strategy Analysis (Airbus)

Baptiste Gillot

December 15, 2025

Abstract

This report presents a quantitative analysis workflow implemented in Python, applied to Airbus (ticker `AIR.PA`). The deliverable follows the same structure as the provided reference report: (i) a single-asset technical-indicator analysis and strategy backtest, and (ii) a multi-asset portfolio analysis with optimization. All interpretations are intentionally omitted; figures produced by the notebook are referenced throughout.

1 Quantitative Trading Strategy Analysis for Airbus Stock

1.1 Analysis Objective

Use historical price data for Airbus to compute technical indicators and build a simple rule-based trading strategy for backtesting.

1.2 Analysis Method

The analysis uses:

- `yfinance` for data collection,
- `pandas` and `numpy` for data processing,
- `matplotlib` for visualization,
- a simple RSI-threshold trading rule for backtesting.

1.3 Core Code Example

1.3.1 Step 1: Data Acquisition and Cleaning

The notebook downloads `AIR.PA` historical data and removes rows containing missing values.

```
import yfinance as yf
import pandas as pd

SYMBOL = "AIR.PA"
START_DATE = "2014-01-01"
END_DATE   = "2024-01-01"

stock_raw = yf.download(SYMBOL, start=START_DATE, end=END_DATE, progress=
    False)
stock = stock_raw.dropna(how="any")
```

1.3.2 Step 2: Moving Averages

The notebook computes and plots two moving averages based on Adjusted Close: MA5 and MA20.

```
stock["MA5"]  = stock["Adj Close"].rolling(window=5).mean()
stock["MA20"] = stock["Adj Close"].rolling(window=20).mean()
```

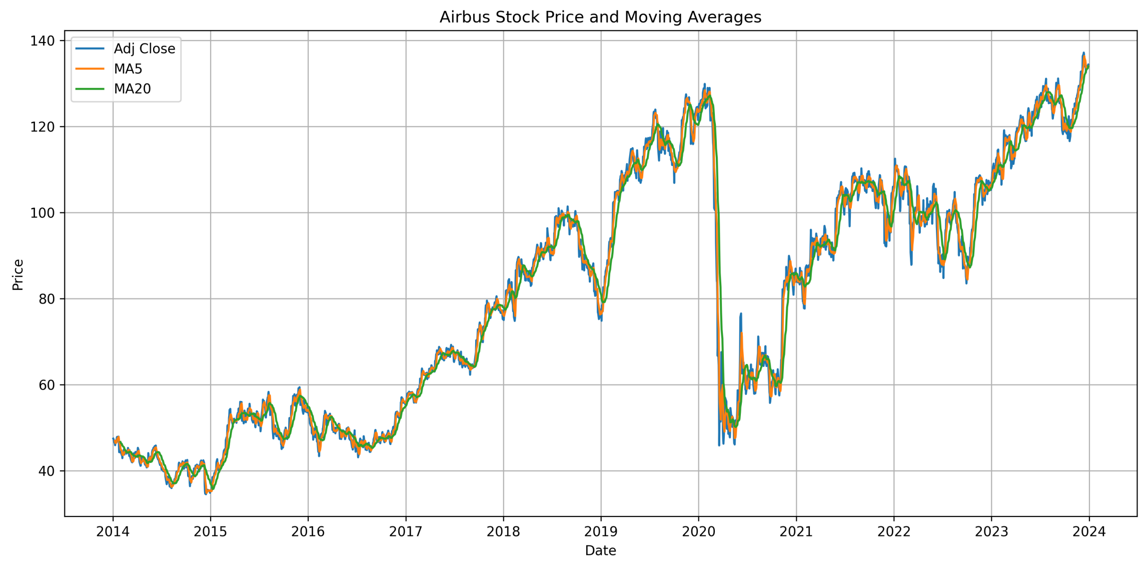


Figure 1: Airbus stock price and moving averages (MA5, MA20).

1.3.3 Step 3: Relative Strength Index (RSI-14)

The notebook computes the RSI with a 14-day rolling window.

```
delta = price.diff()
gain  = delta.clip(lower=0)
loss  = -delta.clip(upper=0)

avg_gain = gain.rolling(window=14).mean()
avg_loss = loss.rolling(window=14).mean()

rs  = avg_gain / avg_loss
rsi = 100 - (100 / (1 + rs))
```

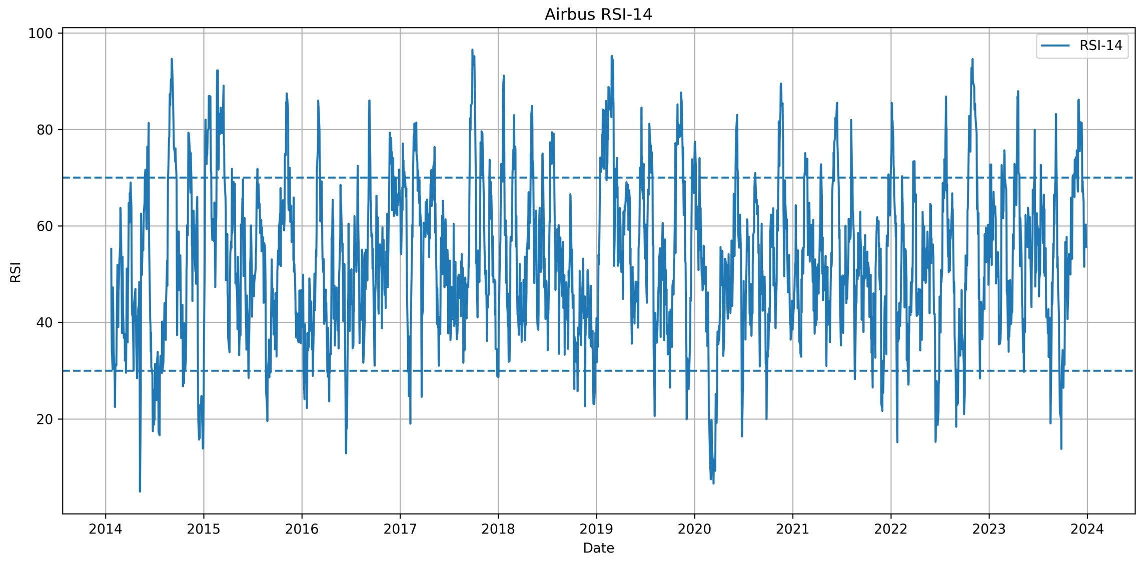


Figure 2: Airbus RSI-14.

1.3.4 Step 4: Strategy Backtesting

The notebook applies a discrete trading rule based on RSI thresholds:

- Buy when RSI < 30,
- Sell when RSI > 70.

The portfolio value is tracked over time (fees and taxes are ignored in the model).

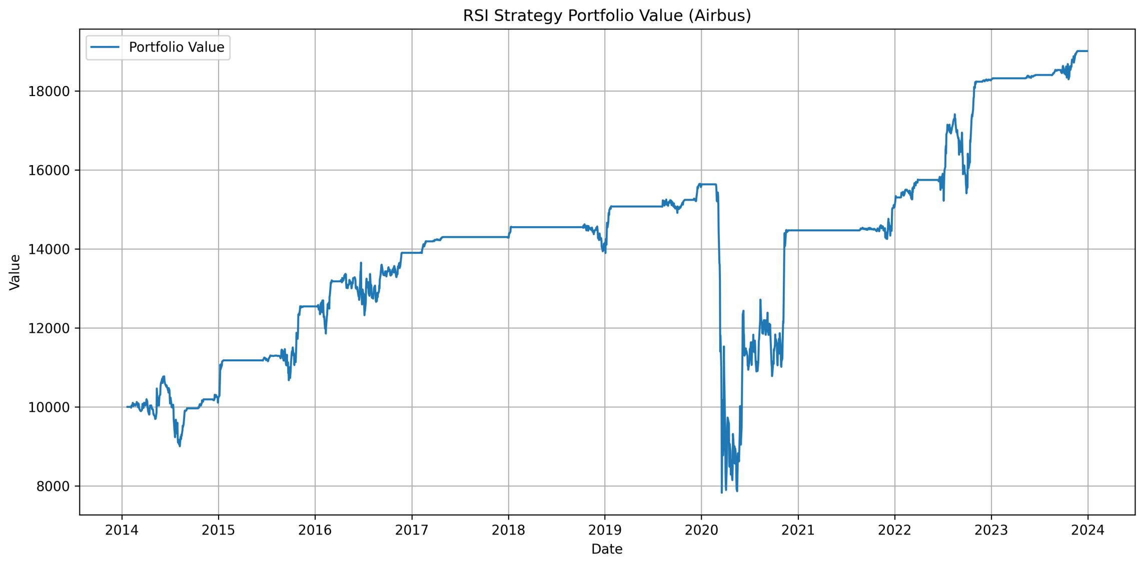


Figure 3: Backtest portfolio value for the RSI threshold strategy on Airbus.

2 Investment Portfolio Optimization Strategy Analysis

2.1 Analysis Objective

Compute returns/risk metrics for a small multi-asset portfolio (including Airbus) and obtain an allocation that maximizes the Sharpe ratio.

2.2 Analysis Method

The notebook:

- downloads Adjusted Close prices for a chosen set of assets,
- computes daily returns, cumulative returns, annualized metrics, and correlations,
- performs Sharpe-ratio optimization with constraints $\sum w_i = 1$ and $0 \leq w_i \leq 1$ using `scipy.optimize`.

2.3 Portfolio Assets

The default asset set in the notebook is:

`AIR.PA` (Airbus), `BA` (Boeing), `SAF.PA` (Safran), `GLD` (Gold ETF).

You can change this list directly in the notebook.

2.4 Key Figures Produced by the Notebook

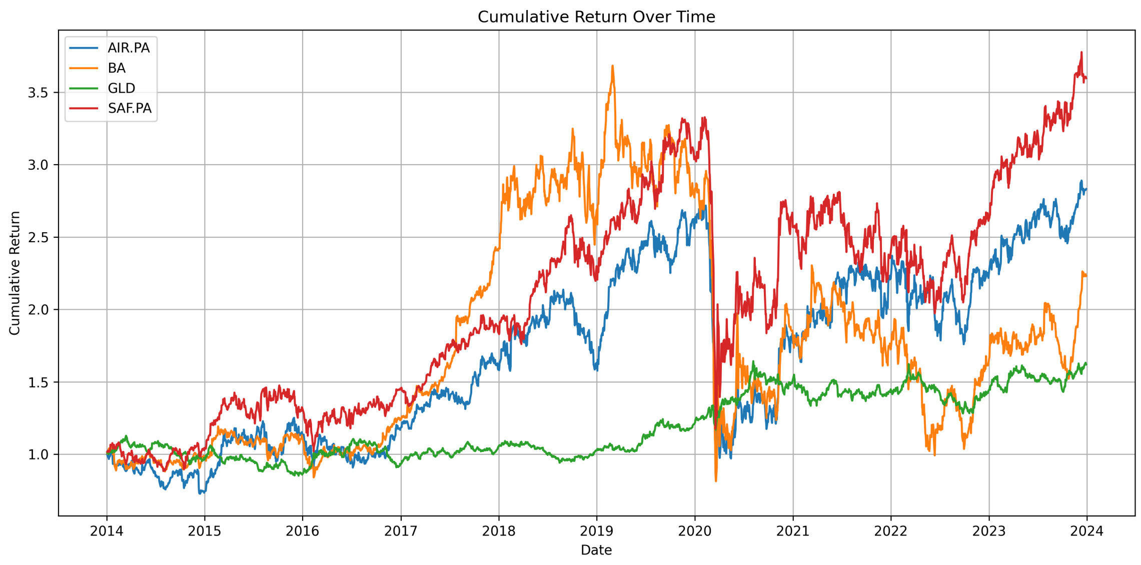
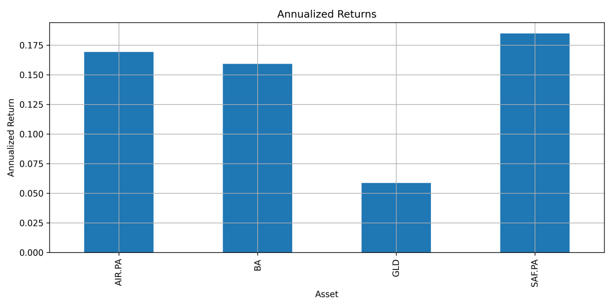
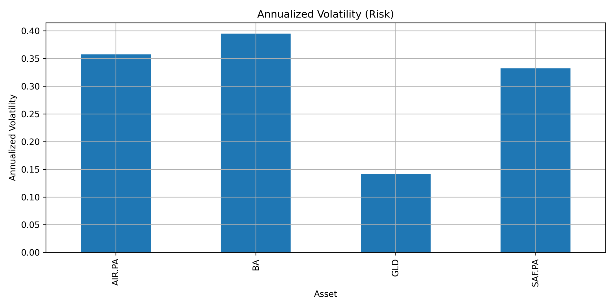


Figure 4: Cumulative returns of the portfolio assets.



(a) Annualized returns.



(b) Annualized volatility (risk).

Figure 5: Annualized metrics for each asset.

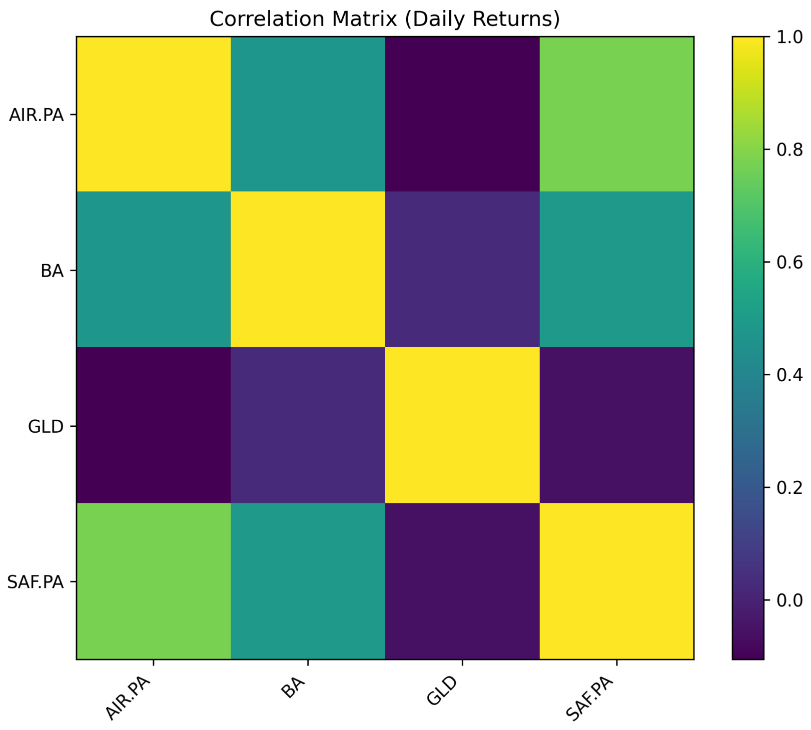


Figure 6: Correlation matrix of daily returns.

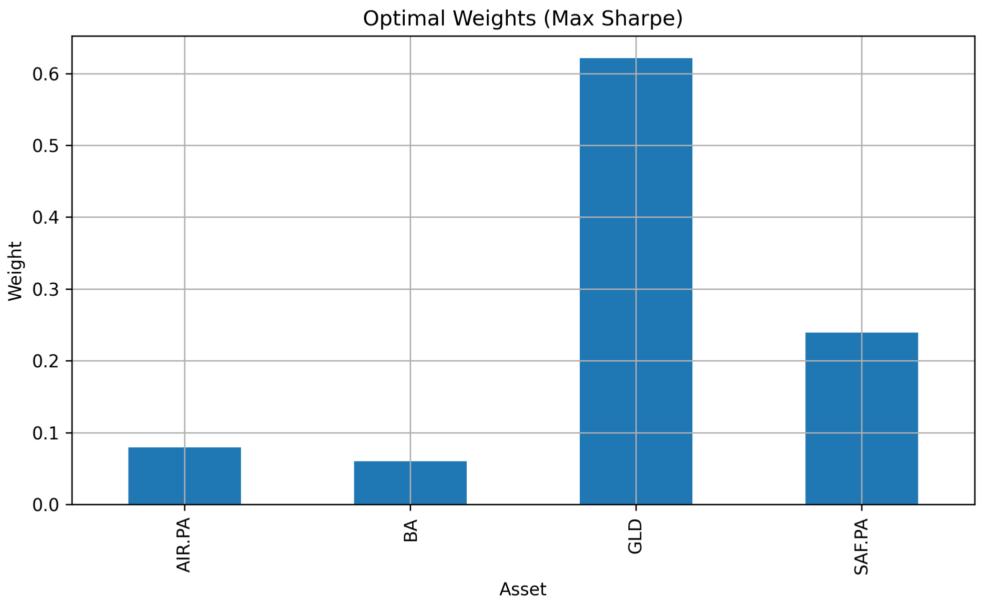


Figure 7: Optimized weights (max Sharpe) under long-only constraints.

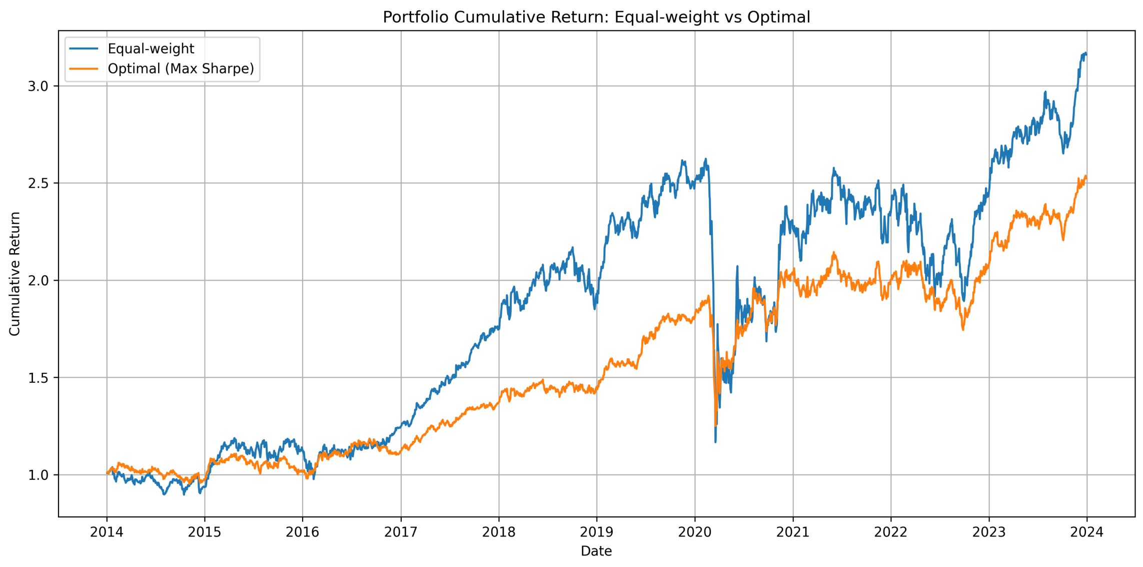


Figure 8: Comparison of cumulative returns: equal-weight vs optimized (max Sharpe) portfolio.

A Reproducibility Notes

- Run the notebook first to generate all figures under `figures/`.
- Then compile this LaTeX file. The report includes the generated images by relative path.
- The notebook also exports:
  - `figures/airbus_stock_data.xlsx`,
  - `figures/portfolio_asset_metrics.csv`,
  - `figures/portfolio_portfolio_metrics.csv`.