Agentic AI Financial Analyst: LLM-Augmented Equity Research and Portfolio Analytics

Andrew Napurano
PACE ID: AN47692N
Course: CS676 Algorithms for Data Science
GitHub: `https://github.com/anapurano211/Agentic-AI-Financial-Analyst`

Abstract

This work presents an *Agentic AI Financial Analyst*, a multi-page Streamlit application that unifies data engineering, classical machine learning (ML), modern portfolio theory, and large language models (LLMs) for equity research. The app ingests fundamentals, technical indicators, and earnings call transcripts; ranks stocks using a weekly logistic regression model; constructs efficient portfolios; and uses LLMs as grounded agents for summarization, sentiment, peer comparison, and risk explanation. We outline the system architecture, the role of LLM-based agents, and the core mathematics behind the optimizer, logistic regression, and tail-risk measures (Value at Risk and Expected Shortfall).

1 System Overview

The platform is a four-page Streamlit app backed by: (i) the Financial Modeling Prep (FMP) API for company fundamentals, technical indicators, and earnings call transcripts; (ii) Yahoo Finance for daily price history; and (iii) Wikipedia-based S&P 500 membership. These sources are merged into a unified analytics table and exposed to both numerical models and LLM agents.

1.1 Screener and Earnings Call Analysis

The first page acts as a screener. Users filter by sector, market cap, and basic valuation metrics (e.g., P/E, P/S, EV/Sales), then drill into a chosen company. For each selected ticker, the app retrieves a company profile and the most recent quarterly earnings call transcript from FMP. An LLM is prompted with the transcript plus recent revenue and EPS growth figures to produce: (1) an overall sentiment label (bullish/neutral/bearish) with polarity and confidence, (2) dimension-level scores for results, guidance, demand, and margins, and (3) a short bullet summary of the main drivers.

Figure 1 shows the earnings analysis panel for Microsoft (MSFT); this is precisely what end users see in the app.

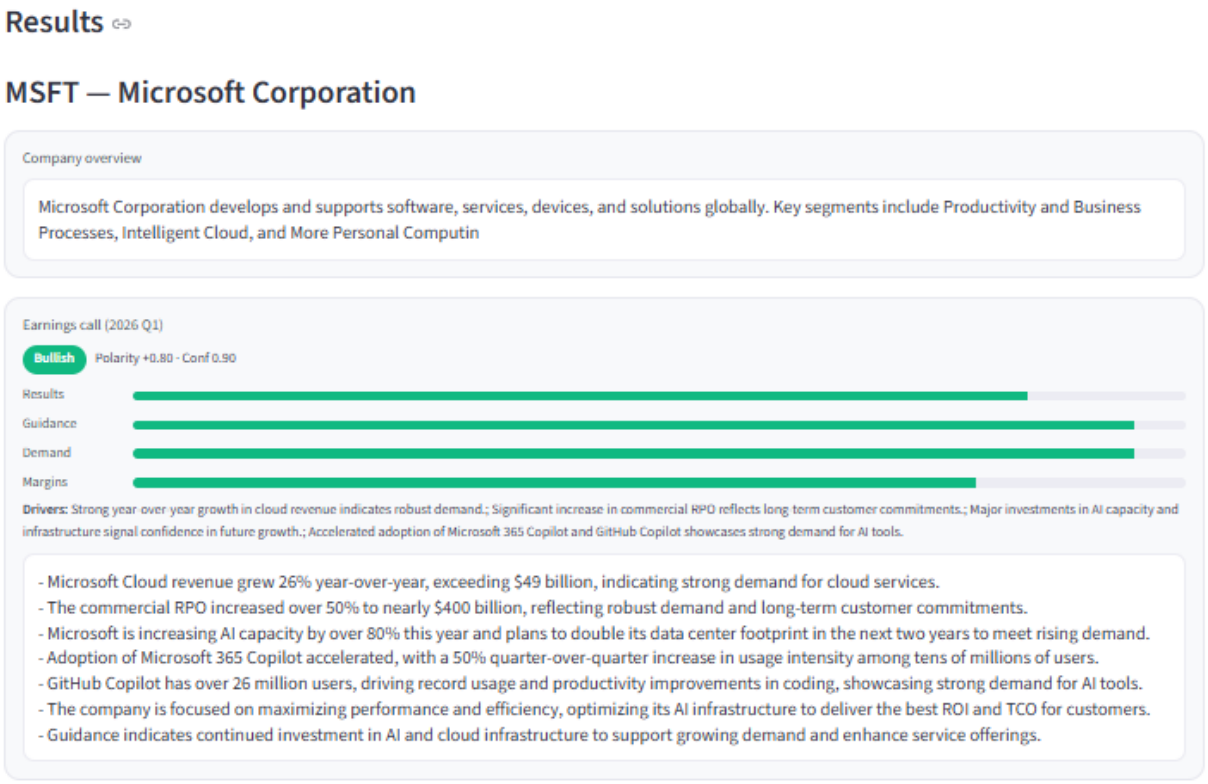

Figure 1: Earnings-call view for Microsoft (MSFT). The panel combines an LLM sentiment label, dimension scores, a driver line, and a bullet summary grounded in both the transcript and recent fundamentals.

1.2 Peer Metrics and Competitive Analysis

The second page aggregates valuation, quality, momentum, and recent performance metrics for a user-defined peer set. The table is passed to an LLM agent that answers questions such as "Which names look like quality at a reasonable price?" or "How does MSFT compare to ORCL on growth and margins?" The agent is restricted to the live DataFrame, preventing hallucinations.

A typical competitive analysis view is shown in Figure 2, with ranking, key ratios, and LLM commentary.

Metric	AAPL	MSFT	NVDA
Operating Margin (%)	31.97	46.27	58.84
Net Margin (%)	26.92	35.71	53.01
ROE (%)	164.05	31.53	103.82
ROA (%)	31.18	16.49	61.56
EPS (TTM)	7.49	13.70	2.97
EPS Growth (Next Year)	9.09 (13.38%)	18.79 (10.49%)	7.54 (61.28%)
Revenue (TTM)	416.16B	281.72B	130.50B
Net Income (TTM)	112.01B	101.83B	72.88B
Revenue Growth (Next Year)	6.82%	15.05%	51.32%

Profitability Insights

1. **Operating Margin**: NVDA leads significantly with an operating margin of **58.84%**, indicating strong cost management and pricing power. MSFT follows at **46.27%**, while AAPL has a lower margin of **31.97%**.
2. **Net Margin**: NVDA also shows the highest net margin at **53.01%**, suggesting superior profitability after all expenses. MSFT and AAPL have net margins of **35.71%** and **26.92%**, respectively.
3. **Return on Equity (ROE)**: AAPL's ROE of **164.05%** is exceptionally high, indicating effective use of equity financing to generate profits. NVDA's ROE is also strong at **103.82%**, while MSFT's is more moderate at **31.53%**.
4. **Return on Assets (ROA)**: NVDA again stands out with an ROA of **61.56%**, showcasing efficient asset utilization. AAPL and MSFT have lower ROAs of **31.18%** and **16.49%**, respectively.

Earnings Growth Projections

- **AAPL** is projected to grow EPS by **13.38%** next year, with a revenue growth of **6.82%**.
- **MSFT** expects an EPS growth of **10.49%** and revenue growth of **15.05%**.
- **NVDA** shows the strongest growth potential, with an EPS growth of **61.28%** and revenue growth of **51.32%**, reflecting robust demand for its products and services.

Figure 2: Competitive analysis view and metrics comparison. The LLM is grounded on the displayed peer table to explain valuation, growth, and quality differences between tickers and to highlight which names combine attractive multiples with strong fundamentals.

1.3 Portfolio Optimizer and Crisis Roles

The third page implements a Monte Carlo efficient frontier and risk diagnostics. Users choose a set of tickers (including LLM-suggested ideas). Daily prices are pulled from Yahoo Finance over a selected window or crisis regime (e.g., 2008–2009, 2020). The app computes daily returns, simulates thousands of random long-only portfolios, and reports annualized return, volatility, Sharpe ratio, max drawdown, and tail risk (VaR and ES) for each portfolio and benchmark.

Figure 3 shows the optimizer summary panel with the user portfolio compared to the max-Sharpe and minimum-volatility portfolios.

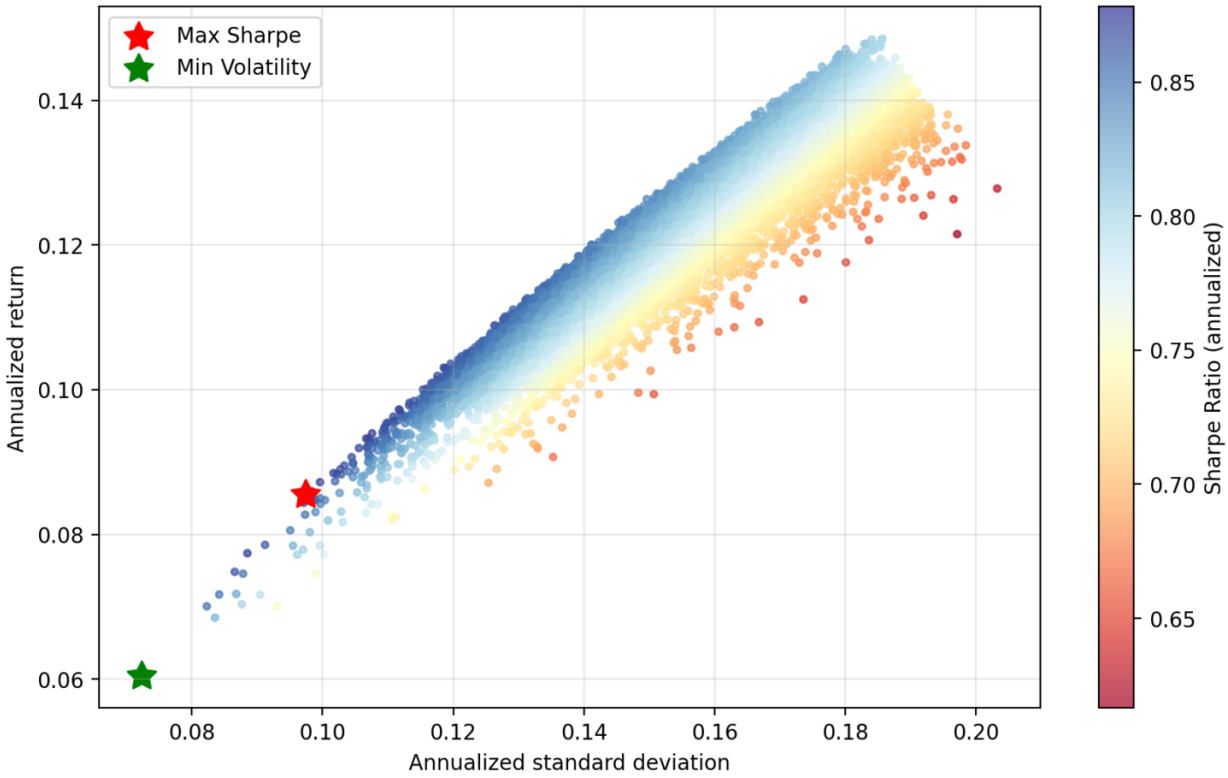

Figure 3: Portfolio optimizer summary. The app reports annualized return, volatility, Sharpe ratio, drawdown, and tail risk for user and model portfolios.

The same returns are used to derive *crisis roles*. For each asset the app constructs a feature vector capturing volatility, drawdown, and correlation behavior across stress windows, and applies K-means clustering. Clusters are labeled as defensive, core, growth, or speculative, and an LLM agent explains each asset's role in plain language. Figure 4 illustrates a crisis-role view.

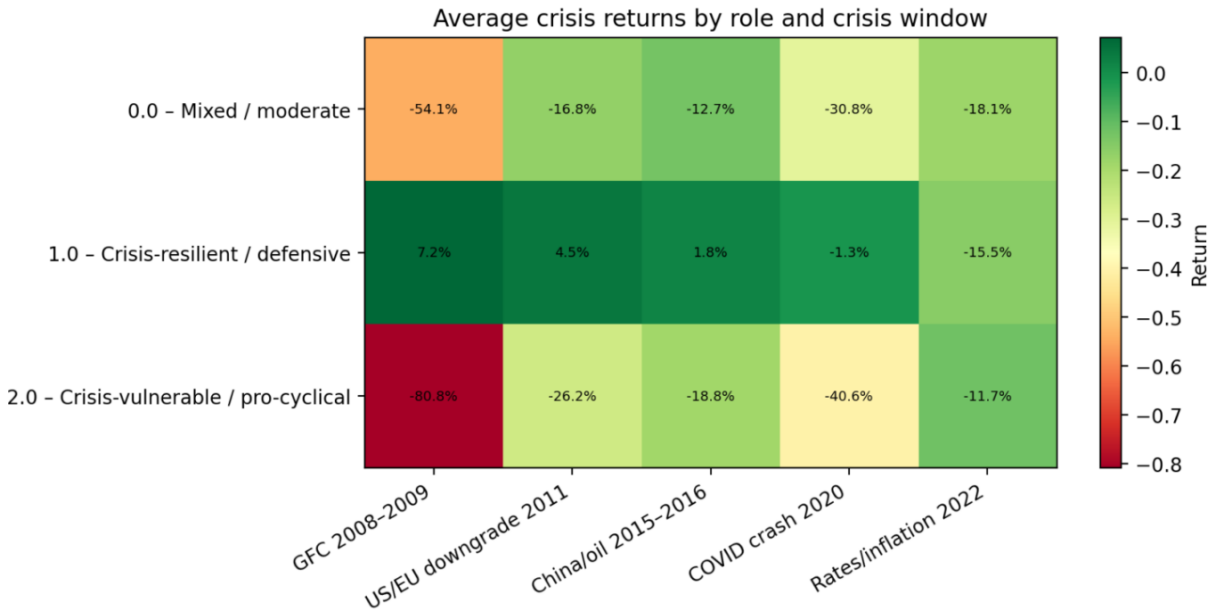

Figure 4: Crisis-role visualization. K-means clusters assets into defensive, core, growth, and speculative roles based on volatility, drawdowns, and correlation patterns, which are then narrated by an LLM.

1.4 Weekly S&P 500 Signals

The final page focuses on weekly ML-based stock selection. For each stock-week in the S&P 500 universe, the app engineers technical features (momentum, moving-average gaps, volatility measures, regime flags) and trains a logistic regression classifier to predict whether a stock will outperform the cross-section in the next week. The page shows the top-k predicted outperformers with probabilities and sectors, classification metrics (accuracy, precision, recall, F1, AUC), and an out-of-sample backtest of the top-k strategy versus VOO.

2 LLM Agents and Grounding

Across pages, LLMs are used in a strictly *grounded* fashion:

- **Earnings agent:** Summarizes transcripts and assigns sentiment using only supplied text and fundamental deltas.
- **Peer agent:** Answers comparative questions using the current peer table as context.
- **Risk agent:** Explains efficient-frontier positions, crisis roles, and tail risk using precomputed statistics.
- **Signal explainer:** Describes why the weekly model prefers certain stocks, referencing their features (e.g. momentum strength, valuation, quality score).

The numerical models (logistic regression, Monte Carlo optimizer, VaR/ES) produce the signals; LLMs only interpret and summarize them.

3 Core Models and Mathematics

We summarize the main quantitative pieces used in the app: logistic regression for weekly signals, modern portfolio theory for the optimizer, and tail-risk measures.

3.1 Weekly Logistic Regression

Each observation corresponds to stock i in week t with feature vector $x_{it} \in \mathbb{R}^d$ and label $y_{it} \in \{0, 1\}$, where $y_{it} = 1$ indicates that the stock outperforms the cross-section in week $t + 1$. The model is

$$\hat{p}_{it} = \mathbb{P}(y_{it} = 1 \mid x_{it}) = \sigma(w^\top x_{it} + b), \qquad \sigma(z) = \frac{1}{1 + e^{-z}}. \tag{1}$$

Given data $\{(x_{it}, y_{it})\}_{i,t}$, parameters (w, b) are fitted by minimizing L2-regularized cross-entropy:

$$\mathcal{L}_\lambda(w, b) = -\sum_{i,t} [y_{it} \log \hat{p}_{it} + (1 - y_{it}) \log(1 - \hat{p}_{it})] + \frac{\lambda}{2}\|w\|_2^2. \tag{2}$$

The app uses a rolling 3-year train / 1-year test scheme to measure generalization. Each week, the model ranks stocks by $\hat{p}_{it}$, and the top-k (e.g. 10% of the universe) form an equal-weight portfolio. Portfolio performance is evaluated versus VOO. This design separates *classification quality* (precision, recall, F1) from *portfolio quality* (alpha, volatility, drawdown).

3.2 Modern Portfolio Theory in the Optimizer

Let n be the number of assets chosen by the user. Daily return vector $r_t \in \mathbb{R}^n$ is computed from prices; $t = 1, \ldots, T$. For a weight vector $w \in \mathbb{R}^n$ with $w_i \geq 0$ and $\mathbf{1}^\top w = 1$, the portfolio return is $r_{p,t} = w^\top r_t$. The sample mean and covariance are

$$\mu = \frac{1}{T}\sum_{t=1}^{T} r_t, \qquad \Sigma = \frac{1}{T-1}\sum_{t=1}^{T}(r_t - \mu)(r_t - \mu)^\top. \tag{3}$$

Expected return and variance of the portfolio are

$$\mu_p(w) = w^\top \mu, \qquad \sigma_p^2(w) = w^\top \Sigma w. \tag{4}$$

Classical Modern Portfolio Theory characterizes the *efficient frontier* as the solution set to

$$\min_w \quad w^\top \Sigma w \tag{5}$$

$$\text{s.t.} \quad w^\top \mu = \mu^*, \quad \mathbf{1}^\top w = 1, \quad w \geq 0, \tag{6}$$

for different targets μ^*. Instead of solving this analytically, the app generates many random feasible w and computes $(\sigma_p(w), \mu_p(w))$, producing a Monte Carlo approximation to the frontier. The annualized Sharpe ratio is

$$\text{Sharpe}(w) = \frac{\mu_p(w) - r_f}{\sigma_p(w)}, \tag{7}$$

with risk-free rate r_f (often set to 0 at daily frequency). The max-Sharpe and minimum-volatility portfolios are the argmax/argmin of Sharpe(w) and $\sigma_p(w)$ over the simulated set. Their statistics appear in the optimizer summary (Figure 3).

3.3 Value at Risk and Expected Shortfall

To capture tail risk, the app computes Value at Risk (VaR) and Expected Shortfall (ES) for each portfolio using historical simulation.

Define portfolio loss as $L_t = -r_{p,t}$ with CDF $F_L(\ell) = \mathbb{P}(L \le \ell)$ and PDF $f_L(\ell)$. For confidence level $\alpha \in (0,1)$, the VaR is the α-quantile:

$$\mathrm{VaR}_\alpha = \inf\{\ell \in \mathbb{R} : F_L(\ell) \ge \alpha\} = F_L^{-1}(\alpha). \tag{8}$$

Expected Shortfall is the expected loss conditional on exceeding VaR:

$$\mathrm{ES}_\alpha = \mathbb{E}[L \mid L \ge \mathrm{VaR}_\alpha] = \frac{1}{1-\alpha}\int_{\mathrm{VaR}_\alpha}^{\infty} \ell\, f_L(\ell)\, d\ell = \frac{1}{1-\alpha}\int_\alpha^1 F_L^{-1}(u)\, du. \tag{9}$$

Empirically, with losses $\{L_t\}_{t=1}^T$:

$$\widehat{\mathrm{VaR}}_\alpha = \mathrm{quantile}_\alpha\{L_1, \ldots, L_T\}, \tag{10}$$

$$\widehat{\mathrm{ES}}_\alpha = \frac{1}{\#\{t : L_t \ge \widehat{\mathrm{VaR}}_\alpha\}} \sum_{t: L_t \ge \widehat{\mathrm{VaR}}_\alpha} L_t. \tag{11}$$

Figure 5 shows a compact risk panel in the app comparing volatility, VaR, and ES for several portfolios and the benchmark.

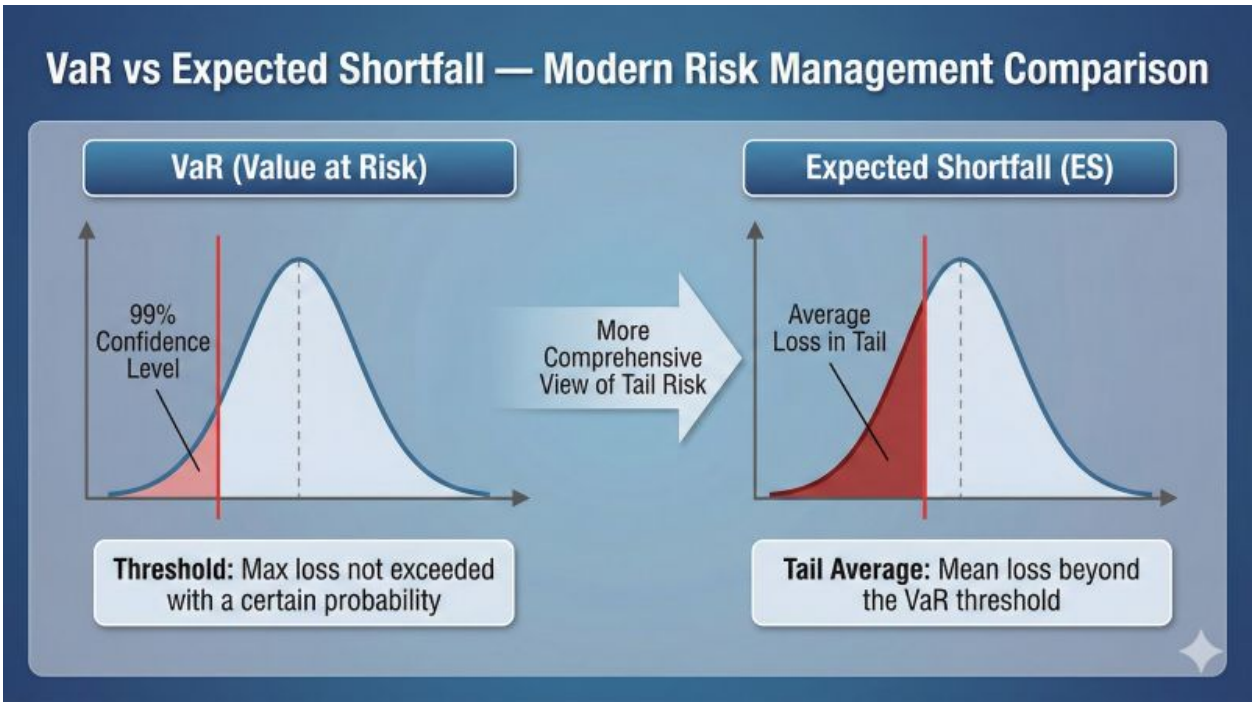

Figure 5: Tail-risk summary. For each portfolio, the app reports volatility, Value at Risk (VaR), and Expected Shortfall (ES), allowing users to compare both average and extreme risk.

4 Conclusion

The Agentic AI Financial Analyst demonstrates how LLMs and traditional quantitative finance can be combined into a single interactive workflow. Logistic regression and portfolio mathematics generate signals and risk profiles; LLM agents, grounded on structured data and transcripts, translate these into human-readable research and explanations. Future work includes richer fundamental factor models, transaction-cost-aware backtests, and expanded retrieval-augmented LLM workflows for deeper, auditably sourced equity research.