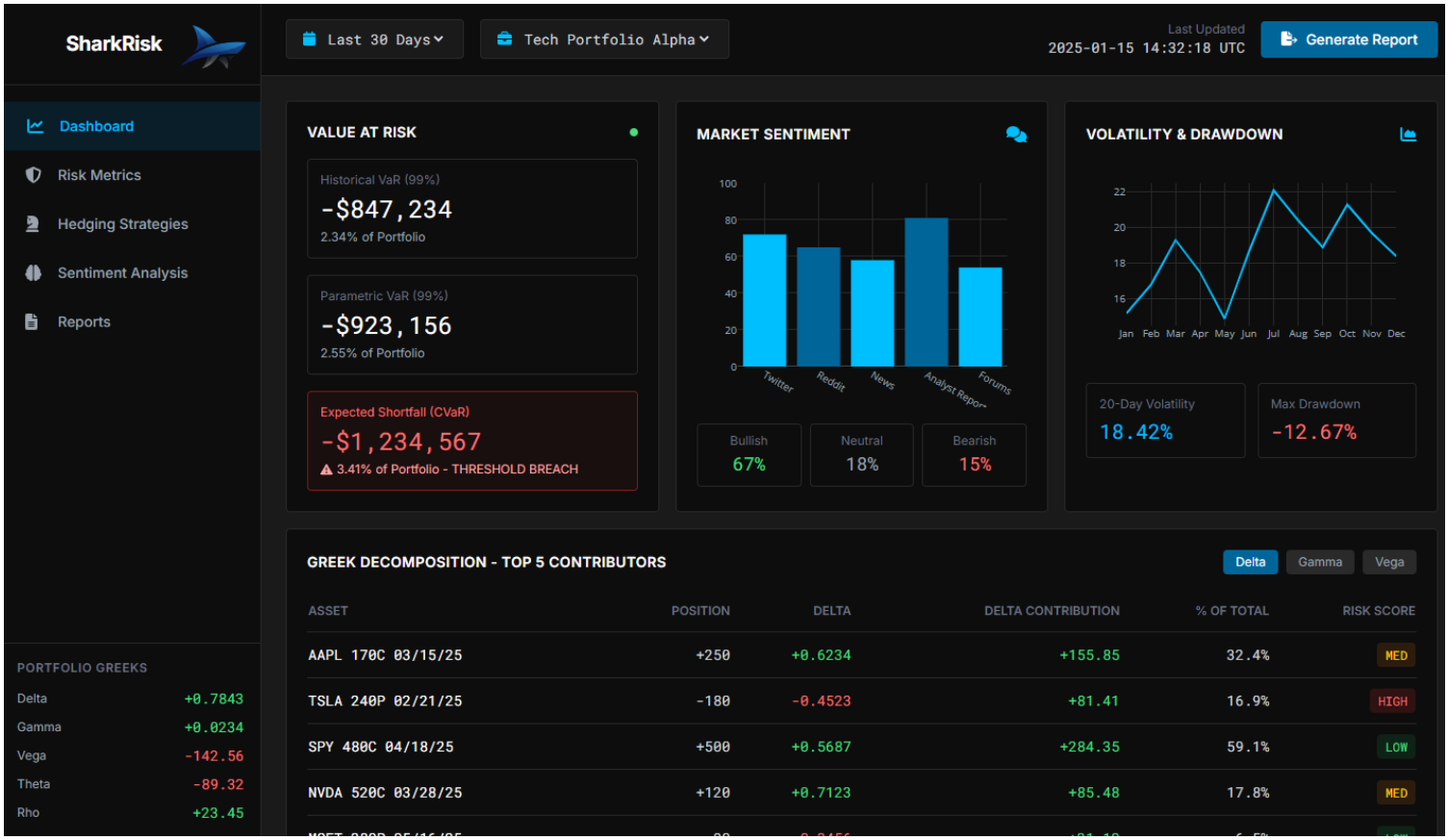
SharkRisk

The first 100% Risk Management Python Library

Risk is not a negative. It's an asset. Manage it that way.



Date: 2025-11-15

Executive Summary

SharkRisk is a full-stack Python portfolio risk analysis solution engineered to transform risk management from a cost center into a strategic value driver. While the manual calculation and integration of complex risk measures (Greeks, VaR, GARCH) remain a significant bottleneck, SharkRisk automates the entire risk management lifecycle, delivering comprehensive risk metrics, AI-driven sentiment analysis, proactive hedging strategies, and compliance-ready automated reports.

Feature	Impact
Comprehensive & Accurate Calculation	Instantly measures Value at Risk (VaR), Credit Exposure, and Greeks (Delta, Gamma, Vega, etc.), going beyond existing partial libraries (e.g., QF-Lib).
AI Sentiment Integration	Incorporates real-time AI/NLP sentiment analysis (Reddit, Twitter) to anticipate market shocks and sentiment-driven volatility spikes.
Proactive Hedging Strategies	Actively proposes targeted hedging methods (options, futures) to neutralize vulnerabilities identified by the risk metrics.
Automated Reporting	Instant generation of complete, transparent, and compliance-ready reports for investors and regulatory institutions.

Context: A Fragmented Market

The modern investment landscape heavily relies on Python for its flexibility and robust quantitative capabilities. However, a significant gap exists between the tooling available for generic Asset Management and specialized Risk Management.

The Python ecosystem is saturated with high-quality libraries focused on portfolio construction, backtesting, and basic financial modelling (Backtrader, Zipline, PyPortfolioOpt, …). While these tools are excellent for constructing and optimizing portfolios, they often treat risk management as a secondary function, providing only high-level metrics like standard deviation or Sharpe Ratios.

In contrast to asset management, the specialized Risk Management space is highly fragmented. While several libraries offer pieces of the puzzle, none provide the complete, end-to-end suite required by institutional risk teams.

- **QuantLib/PyQL:** An industry-standard C++ library with Python bindings. While powerful, its complexity and steep learning curve often limit its use.
- **Riskfolio-Lib:** Primarily focused on portfolio optimization under various risk measures (like CVaR, etc.) but is not designed as a comprehensive risk reporting and hedging platform.
- **PyRisk:** Represents smaller, often specialized, or non-maintained efforts in specific areas of risk modeling.

This fragmentation creates a major operational burden for risk managers and quantitative analysts. To produce a full risk report, a typical risk team is forced to switch between multiple disparate libraries:

1. Use **pandas** to import and clean the data.
2. Use **QuantLib** to calculate complex Greeks for derivatives exposure.
3. Use **Riskfolio-Lib** to optimize portfolio-level CVaR.

4. Use **proprietary code** to calculate Historical VaR and credit exposure.

5. Use **Matplotlib** to generate visualizations.

This reliance on multiple tools leads to Integration Overheads writing glue code and ensuring data structures are compatible, Inconsistency Risk and Inefficiency that prevent real-time risk monitoring and proactive decision-making.

SharkRisk directly addresses this critical need by offering a single, unified, and exhaustive library that integrates all necessary risk functions - from data import and Greek computation to AI sentiment analysis and automated regulatory reporting - into one coherent Python API. This integration eliminates the need to juggle and positions SharkRisk as the definitive full-stack risk management tool.

Core Engine: 5 Main Functions

The SharkRisk library is built around five integrated modules, each designed to automate and enhance a critical stage of the institutional risk management process.

#	Function	Purpose	Value Proposition
1	Import Portfolio	Establishes a clean, validated, and structured dataset for risk analysis.	Ensures data quality, historical depth, and structural validity (weights, tickers) before any computation.
2	Risk Metrics	Calculates a comprehensive suite of portfolio and derivative-level risk measures.	Delivers all critical metrics (VaR, Greeks, Credit Exposure) in one place, moving beyond basic volatility measures.
3	Sentiment Analysis	Leverages AI to incorporate qualitative market factors into risk assessment.	Predicts short-term volatility and risk driven by social media sentiment, adding an alpha factor to risk monitoring.
4	Hedging Strategies	Proposes tactical measures to mitigate identified risks.	Shifts the risk process from passive measurement to proactive, data-driven risk reduction strategies.
5	Automated Reporting	Generates professional reports for investors and compliance bodies.	Streamlines communication and regulatory adherence by instantly translating complex data into transparent reports.

1. import.sharkrisk.py

Objective: To establish a completely clean, validated, and statistically robust dataset, serving as the trusted foundation for all subsequent risk analysis.

Mechanism: The system executes a series of automated checks: File & Format Checks (e.g., mandatory columns), Structural Validation (e.g., weights summing to 100%), Market Data Sourcing (fetching historical time series), and Initial Computations (e.g., Daily Returns and Correlation Matrix).

Output: A validated, analysis-ready portfolio structure that guarantees mathematical soundness and data sufficiency for statistically robust risk metrics.

2. metrics.sharkrisk.py

Objective: To provide a comprehensive, institutional-grade assessment of portfolio and derivative vulnerability across market and non-market risks..

Mechanism: The engine calculates exhaustive metric groups: Loss Quantification (VaR, CVaR, Max Drawdown), Derivative Sensitivity (Greeks: Delta, Gamma, Vega, Theta; and Systemic Risk (Credit Exposure, Correlation). These are augmented by visualizations like Volatility Contribution charts.

Output: Critical loss estimates, essential derivative risk insights, and actionable visual analyses that meet regulatory capital requirements and fund mandates.

3. sentiment.sharkrisk.py

Objective: To provide an early warning system by integrating forward-looking, qualitative market factors into the quantitative risk assessment.

Mechanism: The module acts as an AI Engine using Natural Language Processing (NLP) to scan public Data Sources (e.g., Reddit, Twitter) for portfolio assets. It classifies the discussion context to generate a quantitative Sentiment Score.

Output: A quantitative sentiment score that serves as a unique risk factor input, offering a leading indicator to anticipate sentiment-driven volatility spikes often missed by purely historical models.

4. hedging.sharkrisk.py

Objective: It performs Risk Driver Identification based on contributions from VaR and Greeks. The system then generates Strategy Proposals for specific derivatives (e.g., put options or futures) tailored for Optimization Focus—neutralizing target risk factors while minimizing transaction costs.

Mechanism: The system solves a convex optimization problem to find the optimal hedge quantities that minimize a weighted sum of residual Greek exposures, an Expected Shortfall (ES) penalty, and a linear transaction cost penalty.

Output: Targeted, data-driven solutions for risk reduction that eliminate hedging guesswork and proactively protect capital against adverse market movements.

5. report.sharkrisk.py

Objective: To automate the final, time-consuming step of the risk management cycle, ensuring transparent communication and regulatory adherence.

Mechanism: It creates a Comprehensive Output that automatically gathers charts, KPIs, risk metrics, and hedging recommendations. Reports are generated in a Compliance Ready structure and are easily Customizable for various internal and external stakeholder needs.

Output: Professional, clear, and auditable documentation that streamlines communication for investors and regulators, saving countless hours in manual report assembly.

What We've Already Built

Given the short timeframe for this project, we prioritized solving the single most critical problem for an investment manager: moving from passive risk reporting to active, automated solutions. We observed that most risk platforms only tell you what your risk is (a list of Greeks or a VaR number), leaving you to figure out the solution. We chose to build the solution first.

Our core engine implements a **Constrained Hedge Optimizer**. This module first calculates the precise Greek sensitivities (Delta, Gamma, Vega) of the entire portfolio and a universe of available hedges. It then uses a powerful optimization library, **cvxpy**, to solve a complex problem: What is the cheapest and most efficient set of trades to neutralize our unwanted exposures, while also respecting tail risk and transaction cost constraints?

The outcome is not a theoretical number but a clear, actionable trade plan. The system automatically generates a final, rounded list of trades, such as **+7 AAPL shares** or **-17 AAPL CALL ATM**, that are ready for execution. This demonstrates a complete, end-to-end pipeline from complex analysis directly to a practical, automated, and intelligent hedging decision, proving the value of our focused approach.

Our Innovative Edge: The AI-Driven Sentiment Engine

Our vision extends beyond traditional, backward-looking risk models. We recognize that quantitative metrics like VaR or volatility are often lagging indicators; they only tell you about a risk after it has already impacted the market. To solve this, our most innovative planned module, sentiment.sharkrisk.py, is designed as a forward-looking early warning system, moving from reactive to predictive risk management.

This module will function as a dedicated AI Engine, leveraging modern Natural Language Processing (NLP). It is being designed to continuously scan and ingest high-velocity, unstructured data from public sources like Reddit, Twitter, and financial news feeds. It won't just count keywords; it will analyze the context and nuance of the discussion around the specific assets in a portfolio, classifying this chatter to generate a single, quantitative Sentiment Score.

The output is the real innovation: a unique risk factor that is entirely independent of historical price data. This score will act as a **leading indicator**, providing the first measurable signal of a potential sentiment-driven volatility spike. By integrating this qualitative, forward-looking data directly into our quantitative assessment, we will be able to anticipate and hedge systemic risks that purely historical models would be blind to until it's too late.

We didn't build this module yet for one simple reason: it is 100% dependent on paid API access to data providers like Twitter/X and Reddit. Given our limited time, we made the practical decision to first build and perfect the core SharkRisk optimization engine, which can run entirely offline. The sentiment module is the clear next step, and its development will begin as soon as we secure that API access.

Business Model

SharkRisk utilizes a tiered subscription model to target institutional clients, ensuring scalable, reliable revenue. The Pro tier provides a clear path to scaling revenue as institutional usage is standardized under a predictable monthly fee. The model's value proposition is the replacement of fragmented, high-overhead workflow solutions with a single, unified API.

The financial viability is established by relatively low fixed costs necessary for a software library. Upfront Development Costs primarily covered the initial integration of specialized data feeds, cloud development, and the labor required to build the core VaR/Greeks and NLP engines. Ongoing Monthly Operating Costs are dominated by external institutional data subscriptions and cloud hosting expenses for continuous risk computations.

The model achieves high margins quickly because the marginal cost of serving an additional client is near zero. This high leverage is validated by a clear target: securing just 50 Pro clients allows the platform to reach a net monthly profit exceeding $20,000, demonstrating a strong financial path.

Future Improvements

Future development is focused on two areas. First, enhancing **Derivatives Data Importation** to support complex derivative fields (Strike, Expiry, Type) for accurate non-linear risk modeling. Second, implementing **Seamless Integration with the Bloomberg Terminal App (BLPAPI)** to access institutional-grade, real-time data, which is crucial for enabling true intra-day risk monitoring.

References

Click the reference to get access

- Mastering the Black-Scholes Model with Python: A Comprehensive Guide to Option Pricing
- Option Greeks by Analytic & Numerical Methods with Python
- pygreeks
- Price options using Black-Scholes in Python
- VinegarHill-FinanceLabs
- Option Payoffs, Black-Scholes, and the Greeks, KidQuant
- QuantLib
- pyRisk
- Riskfolio-lib
- PyPortfolioOpt

Access to the code

The code represents an experimentation in implementing financial derivatives and interfaces in Python. Its development utilized provided course materials, external references, and generative AI tools.

Please note the dashboard overview illustrates the forecasted final version of the tool.

Click to access a first version of the SharkRisk code, the report generation jupyter notebook code. Each file can be found in the repository of the project