Crypto Signal Generator (Python)

Technical Report

Francisco Moya

Contents

Chapter 1

Introduction

The cryptocurrency market is characterized by extreme volatility, regime shifts, and microstructural frictions that make traditional, deterministic trading rules insufficient. Traders and analysts require tools that are not only able to process real-time data efficiently but also capable of quantifying uncertainty and adapting to changing conditions. In this context, probabilistic frameworks offer a natural advantage: instead of issuing binary "buy/sell" recommendations, they provide distributions over possible future outcomes, allowing for more informed decision-making.

The **Crypto Signal Generator (Python)** was designed to address this need. Unlike conventional screeners or indicator-based systems, it integrates statistical modeling, volatility dynamics, and cost-aware risk management into a unified pipeline. The framework computes log returns and volatility features from Binance market data, models conditional regime transitions through Markov chains, and propagates these probabilities across multiple horizons to produce fan charts and predictive intervals. This multi-horizon perspective enhances interpretability and robustness, as it reveals not just a single forecast but a spectrum of possible scenarios.

The system also incorporates a decision layer that translates probabilistic forecasts into actionable signals. This layer adjusts for transaction costs and market noise using cost-aware thresholds and hysteresis, while position sizes are determined by a trimmed Kelly strategy that adapts to volatility regimes. In addition, the design includes contextual models from option pricing and volatility theory—such as Black–Scholes, Heston, and SVI parametrizations—both as academic references and as benchmarks for validating empirical features.

From a methodological perspective, the contribution of this project lies in its combination of:

1. **Feature Engineering:** robust return and volatility measures, normalized for stability.

2. **Probabilistic Modeling:** Markov chains with conditional transitions and entropy-based uncertainty metrics.

3. **Forecasting:** multi-horizon propagation yielding predictive intervals.

4. **Risk-Aware Decisions:** signals and position sizing grounded in probabilities, volatility, and costs.

5. **Validation:** probabilistic scoring rules and portfolio-level performance metrics.

Overall, the Crypto Signal Generator represents a modular, interpretable, and practical framework that bridges academic rigor with real-world applicability. It transforms raw market data into structured probabilistic insights, enabling users to explore scenarios, assess risks, and derive executable trading decisions.

Chapter 2

Objectives

The primary objective of the **Crypto Signal Generator (Python)** is to provide a probabilistic and risk-aware framework for cryptocurrency trading analysis. Instead of relying on deterministic signals or single-point forecasts, the system is designed to output probability distributions, predictive intervals, and cost-adjusted strategies that improve both interpretability and robustness.

The specific objectives can be summarized as follows:

1. **Real-Time Probabilistic Forecasting**
   - Compute forward-looking distributions of market states (e.g., bearish, neutral, bullish) using Markov chain models.
   - Propagate probabilities across multiple horizons to generate fan charts and conditional predictive intervals.
2. **Feature-Rich Volatility and Market Microstructure Analysis**
   - Extract and process log returns, realized volatility, EWMA volatility, and ATR.
   - Normalize features through z-scores for stability across different market conditions.
   - Incorporate microstructure features such as spreads, order book imbalance, funding rates, and open interest.
3. **Risk-Aware Signal Generation**
   - Translate probabilistic forecasts into actionable signals through cost-aware thresholds.
   - Apply hysteresis rules to avoid excessive signal flipping.
   - Implement trimmed Kelly position sizing, adjusted for volatility regimes and practical constraints.
4. **Contextual Integration with Volatility and Option Models**
   - Use classical option pricing models (Black–Scholes, Heston, jump-diffusion) as academic benchmarks for volatility behavior.

- Leverage implied volatility surfaces (SVI parametrizations) as external validation for realized volatility measures.

5. **Validation and Performance Measurement**

   - Evaluate probabilistic accuracy with proper scoring rules (Brier score, log-loss).
   - Assess portfolio-level performance using financial metrics such as Sharpe ratio and drawdown.
   - Ensure that predictions remain both statistically consistent and economically meaningful.

6. **Modularity and Extensibility**

   - Design the system as a modular Python framework, allowing future integration of additional features, models, or assets.
   - Maintain interpretability and transparency, ensuring that each component can be traced back to its mathematical foundation.

Chapter 3

System Overview

The **Crypto Signal Generator (Python)** is structured as a modular pipeline that transforms raw market data into probabilistic forecasts, predictive intervals, and actionable trading signals. Each stage is designed to be interpretable and extensible, ensuring that the system can evolve as new data sources and modeling approaches are integrated.

The pipeline consists of five main layers:

1. **Data Ingestion and Feature Engineering**
    - Market data is retrieved from Binance, including OHLCV series, order book depth, funding rates, and open interest.
    - Features such as log returns, realized volatility, EWMA volatility, ATR, and normalized z-scores are computed.
    - Microstructure indicators, including relative spreads and order book imbalance, complement the feature set.
2. **Probabilistic Core (Markov Chains)**
    - Market dynamics are modeled as transitions between discrete states (e.g., strong downtrend, mild downtrend, neutral, mild uptrend, strong uptrend).
    - Transition probabilities are estimated using maximum likelihood with Bayesian smoothing, ensuring robustness in low-sample regimes.
    - Conditional Markov chains incorporate feature dependencies, making state transitions responsive to volatility and other signals.
    - Entropy metrics quantify predictive uncertainty and confidence levels.
3. **Multi-Horizon Forecasting**
    - Transition probabilities are propagated over multiple steps, producing forecasts across different horizons (minutes, hours, or days).
    - From these forecasts, conditional means, variances, and fan charts are generated, illustrating the range of likely price outcomes.
    - Lognormal approximations are used to translate return distributions into price intervals.
4. **Signal and Risk Layer**

- Forecast probabilities are mapped into actionable signals by evaluating the probability of upward vs. downward moves.
- Cost-aware thresholds account for trading fees and slippage.
- Hysteresis is applied to prevent excessive trading caused by small fluctuations in probabilities.
- Position sizing follows a trimmed Kelly criterion, adapting to current volatility and risk constraints.

5. **Validation and Output**

   - Forecasts and signals are validated using both probabilistic scoring rules (e.g., Brier score, log-loss) and portfolio metrics (e.g., Sharpe ratio, drawdown).
   - Outputs include probability dashboards, fan charts, and logs of simulated trading signals with performance evaluation.
   - The modular architecture allows integration of contextual layers, such as volatility surface analysis (SVI) and option pricing benchmarks.

This layered architecture ensures a clear separation of concerns: data processing, probabilistic modeling, forecasting, decision-making, and validation. It reflects a balance between theoretical rigor and practical usability, bridging academic models with real-world trading constraints.

Chapter 4

Data and Feature Engineering

The first step in the Crypto Signal Generator is the construction of a robust feature set from raw Binance market data. These features capture both return dynamics and volatility structure, while also integrating microstructural signals such as spreads and order book imbalance. By combining traditional time-series indicators with microstructure data, the system ensures that the probabilistic core is informed by both macro-level trends and market frictions.

4.1 Logarithmic Returns

Log returns provide a scale-invariant and additive measure of price changes. For price series p_t:

$$r_t = \ln \frac{p_t}{p_{t-1}}$$

This transformation stabilizes variance and ensures that multi-period returns can be aggregated additively, which is convenient for Markov modeling and multi-horizon forecasting.

4.2 Realized Volatility

Realized volatility measures short-term variability of returns within a given window N:

$$\sigma^2_{RV,t} = \frac{1}{N-1} \sum_{j=1}^{N} (r_{t-j} - \bar{r})^2$$

This metric provides a non-parametric estimate of market turbulence and is particularly useful in distinguishing between calm and stressed regimes.

4.3 EWMA Volatility (RiskMetrics)

The exponentially weighted moving average (EWMA) is used to emphasize recent returns:

$$\sigma_t^2 = (1-\lambda) r_t^2 + \lambda \sigma_{t-1}^2, \quad \lambda \in [0.9, 0.99]$$

This formulation adapts quickly to volatility clustering while smoothing out noise. It is a cornerstone of the RiskMetrics framework and aligns well with regime-based transitions.

4.4 Average True Range (ATR)

ATR captures intraday price range dynamics and is sensitive to volatility spikes:

$$ATR_t = \frac{1}{n} \sum_{i=1}^{n} \max\{H_i - L_i, |H_i - C_{i-1}|, |L_i - C_{i-1}|\}$$

where H_i, L_i, C_i represent the high, low, and close prices. This metric complements return-based volatility with range-based information.

4.5 Normalized Z-Scores

To stabilize inputs across different volatility regimes, features are standardized:

$$u_t = \frac{r_t}{\sigma_t + \varepsilon}$$

This normalization ensures that the probabilistic core does not overweight extreme return values due to scale differences.

4.6 Microstructure Features

4.6.1 Relative Spread

$$\text{Spread}_t = \frac{Ask_t - Bid_t}{(Ask_t + Bid_t)/2}$$

Provides a measure of transaction cost pressure and liquidity.

4.6.2 Order Book Imbalance

$$\text{Imb}_t = \frac{\sum q_{bid} - \sum q_{ask}}{\sum q_{bid} + \sum q_{ask}}$$

Captures buying vs. selling pressure at the microstructural level.

4.6.3 Funding Rates and Open Interest

- **Funding Rates:** Indicate directional bias in perpetual futures markets.
- **Open Interest (OI):** Reflects capital commitment and potential leverage buildup.

Together, these features form a multi-scale representation of the market, combining short-term volatility, medium-term ranges, and structural liquidity signals. This enriched feature set feeds directly into the Markov chain core, allowing the probabilistic model to capture regime dynamics conditioned on observable market behavior.

Chapter 5

Probabilistic Core: Markov Chains

At the heart of the Crypto Signal Generator lies a **Markov chain framework**, which models market dynamics as transitions between discrete states. Instead of predicting a single deterministic outcome, the system assigns probabilities to future regimes and propagates them forward in time. This probabilistic core ensures interpretability while capturing the persistence and clustering properties of financial time series.

5.1 Market States

The market is represented by a finite set of discrete states, for example:

- Strong Downtrend (S1)
- Mild Downtrend (S2)
- Neutral (S3)
- Mild Uptrend (S4)
- Strong Uptrend (S5)

This discretization allows mapping of continuous return and volatility dynamics into interpretable regimes.

5.2 Transition Probabilities

The fundamental object is the transition matrix P, where each element represents the probability of moving from state i to state j:

$$P_{ij} = \Pr(S_{t+1} = j \mid S_t = i)$$

The one-step forecast is given by:

$$\pi_{t+1} = \pi_t P$$

where π_t is the distribution over states at time t.

5.3 Maximum Likelihood Estimation with Smoothing

Transition probabilities are estimated from observed state transitions using maximum likelihood. To prevent zero probabilities (common in sparse data), Bayesian smoothing is applied:

$$\widehat{P}_{ij} = \frac{N_{ij} + \alpha}{\sum_j N_{ij} + K\alpha}$$

where N_{ij} is the count of transitions from i to j, K is the number of states, and α is a smoothing parameter.

5.4 Multi-Step Forecasting

By raising the transition matrix to higher powers, the system generates multi-horizon forecasts:

$$\pi_{t+h} = \pi_t P^h$$

This property makes Markov chains particularly suitable for horizon-dependent predictions.

5.5 Stationary Distribution

The stationary distribution represents the long-run equilibrium probabilities of being in each state:

$$\pi^* = \pi^* P, \quad \sum_i \pi_i^* = 1$$

This distribution provides insights into the persistent characteristics of the market.

5.6 Conditional Markov Chains with Features

To make the system responsive to market conditions, transition probabilities are conditioned on features such as volatility or order book imbalance. This is achieved via a logistic parametrization:

$$\Pr(S_{t+1} = j \mid S_t = i, x_t) = \frac{e^{\beta_{ij}^\top x_t}}{\sum_\ell e^{\beta_{i\ell}^\top x_t}}$$

Here, x_t represents feature vectors (returns, volatility, spreads), and β_{ij} are parameters linking features to transition dynamics.

5.7 Entropy as Uncertainty Measure

Predictive entropy is used to quantify confidence in the forecasted distribution:

$$H(\pi_{t+1}) = -\sum_j \pi_{t+1,j} \ln \pi_{t+1,j}$$

- Low entropy: one regime dominates $\rightarrow$ high confidence forecast.
- High entropy: distribution spread across states $\rightarrow$ high uncertainty.

Entropy provides a diagnostic tool to assess when signals are reliable versus when caution is warranted.

Chapter 6

Multi-Horizon Forecasting

While one-step transition probabilities provide valuable short-term insights, effective trading and risk management require a forward-looking view across multiple horizons. The Crypto Signal Generator extends the Markov chain framework to propagate state probabilities several steps ahead, enabling the construction of **fan charts**, conditional expectations, and predictive intervals for future returns and prices.

6.1 Multi-Step Probability Propagation

Using the Markov property, the distribution of states at horizon h is obtained by repeated application of the transition matrix:

$$\pi_{t+h} = \pi_t P^h$$

This propagation allows the system to compute forward distributions not only for the next step but also for medium- and longer-term horizons.

6.2 Conditional Mean of Cumulative Returns

The expected cumulative return over a horizon H is:

$$\mathbb{E}[R_H] = \sum_{h=1}^{H} \sum_{k} \pi_{h,k}\, \mu_k\, \sigma_t$$

where μ_k represents the mean return of state k, and σ_t scales the contribution by current volatility.

6.3 Conditional Variance of Cumulative Returns

Similarly, the horizon-dependent variance is approximated as:

$$\mathbb{V}[R_H] \approx \sum_{h=1}^{H} \sum_{k} \pi_{h,k}\, v_k\, \sigma_t^2$$

where v_k is the variance associated with state k. This quantifies uncertainty accumulation across multiple steps.

6.4 Predictive Intervals (Lognormal Approximation)

To translate return distributions into price forecasts, a lognormal approximation is applied:

$$\hat{P}^q_{t+h} = P_t \exp\Big(\mathbb{E}[R_h] \pm z_q \sqrt{\mathbb{V}[R_h]}\Big)$$

where z_q is the quantile of the standard normal distribution (e.g., $z_{0.975}$ for a 95% confidence band).

6.5 Fan Charts

The combination of conditional means, variances, and predictive intervals produces fan charts that visualize the widening range of possible outcomes as the horizon increases. These charts provide an intuitive view of:

- Central tendencies (expected path).
- Uncertainty bands (confidence intervals).
- Asymmetry between upside and downside scenarios.

Fan charts thus serve as a powerful visualization of market uncertainty, enabling risk-aware decision-making rather than over-reliance on single-point forecasts.

Chapter 7

Signals and Risk Management

The ultimate goal of the Crypto Signal Generator is to transform probabilistic forecasts into **actionable trading signals** while ensuring that risk is properly managed. This is achieved through a layered decision process: (1) extracting directional probabilities, (2) applying cost-aware thresholds, (3) filtering signals with hysteresis, and (4) determining position size via a trimmed Kelly criterion.

7.1 Probabilities of Upward and Downward Moves

From the multi-horizon distribution, upward and downward move probabilities are aggregated:

$$p_t^{UP} = \pi_{t+1,4} + \pi_{t+1,5}, \quad p_t^{DN} = \pi_{t+1,1} + \pi_{t+1,2}$$

where states 4–5 correspond to upward regimes, and states 1–2 to downward regimes. These probabilities form the basis of trading decisions, indicating the likelihood of price appreciation or depreciation.

7.2 Cost-Aware Thresholds

Trading signals must overcome transaction costs and slippage. A dynamic threshold is introduced:

$$\tau_t = \frac{c_{round}}{k\sigma_t} + \tau_0$$

where c_{round} is the round-trip cost, σ_t is current volatility, k is a scaling factor, and τ_0 is a baseline probability threshold. This ensures that only signals with sufficient expected edge relative to costs are acted upon.

7.3 Hysteresis Filter

To avoid excessive flipping between long and short positions due to small probability changes, hysteresis rules are applied:

$$\text{Open long if } p_t^{UP} > \tau_t + \delta$$

$$\text{Close or avoid long if } p_t^{UP} < \tau_t - \delta$$

where δ is a buffer parameter controlling signal persistence. This reduces noise and stabilizes trading decisions.

7.4 Position Sizing: Trimmed Kelly Criterion

Position size is determined using a modified Kelly formula, adapted for volatility targeting and practical constraints:

$$w_t = \text{clip}\left(\kappa \cdot \frac{p_t^{UP} - p_t^{DN}}{\tau_t} \cdot \frac{\sigma_{target}}{\sigma_t} \cdot \frac{1}{1+g_t},\ 0,\ w_{max}\right)$$

where:

- κ = Kelly scaling factor (to trim aggressiveness).
- σ_{target} = desired portfolio volatility.
- g_t = leverage or risk adjustment factor.
- w_{max} = maximum allowable position size.

This formula ensures that risk is dynamically adjusted to both volatility regimes and signal confidence.

7.5 Practical Signal Output

The decision layer produces signals of the form:

- **Direction:** Long, Short, or Neutral.
- **Confidence:** Probability-based.
- **Size:** Fraction of capital allocated.

Signals are logged alongside probabilities, thresholds, and risk metrics to enable both real-time monitoring and retrospective evaluation.

Chapter 8

Volatility and Jump Models (Contextual Layer)

While the Crypto Signal Generator primarily relies on empirical features and Markov-based probabilistic modeling, theoretical models of volatility and jumps provide an important contextual layer. These models serve as academic benchmarks and validation tools, helping to interpret observed market dynamics and to cross-check the plausibility of volatility estimates derived from market data.

8.1 Black–Scholes Model (Baseline)

The Black–Scholes model assumes constant volatility and lognormal price dynamics:

$$C = S_0 N(d_1) - K e^{-rT} N(d_2)$$

with

$$d_1 = \frac{\ln(S_0/K) + (r + \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}, \quad d_2 = d_1 - \sigma\sqrt{T}$$

Although unrealistic in practice, Black–Scholes provides a foundational reference for option pricing and implied volatility extraction.

8.2 Heston Model (Stochastic Volatility)

The Heston model introduces mean-reverting stochastic variance:

$$\begin{aligned} dS_t &= \mu S_t dt + \sqrt{v_t} S_t dW_t^{(1)} \\ dv_t &= \kappa(\theta - v_t)dt + \xi\sqrt{v_t} dW_t^{(2)} \end{aligned}$$

This captures volatility clustering observed in crypto markets, where variance is itself random and persistent.

8.3 Merton Jump-Diffusion Model

Merton extends Black–Scholes by incorporating lognormal jumps:

$$\frac{dS_t}{S_t} = (\mu - \lambda k)dt + \sigma dW_t + dJ_t$$

where jumps occur with intensity λ, and J_t represents the jump component. This aligns with the frequent sudden shocks in crypto markets.

8.4 Kou Double-Exponential Jump Model

Kou refined the jump structure by modeling the jump distribution as double-exponential:

$$J \sim \mathrm{DE}(\eta_1, \eta_2)$$

This allows for asymmetric jump behavior, capturing the fatter tails and skew often observed in Bitcoin and other cryptocurrencies.

8.5 Bates Model (Heston + Jumps)

The Bates model combines stochastic volatility (Heston) with jumps:

$$dS_t = S_t(\mu dt + \sqrt{v_t} dW_t^{(1)} + dJ_t)$$

This provides the most realistic representation of markets where both stochastic volatility and discontinuous jumps coexist.

8.6 Role in the Framework

In this project, these models are not calibrated in real-time but rather serve as:

- **Benchmarks** to validate volatility estimates (realized vs. theoretical).
- **Contextual references** when interpreting implied volatility surfaces.
- **Academic grounding** to connect empirical methods with established option pricing theory.

Chapter 9

Implied Volatility and Smiles

Implied volatility (IV) derived from option prices provides a forward-looking measure of market expectations. Unlike realized volatility, which is backward-looking, implied volatility reflects the market's consensus about future uncertainty. In the context of the Crypto Signal Generator, volatility smiles and surfaces act as an external validation layer, offering benchmarks against which empirical volatility estimates can be compared.

9.1 SVI Parametrization

The Stochastic Volatility Inspired (SVI) model provides a flexible parametric form for the implied variance as a function of log-moneyness k:

$$w(k) = a + b\left(\rho(k-m) + \sqrt{(k-m)^2 + \sigma^2}\right)$$

where:

- a: overall level of variance,
- b: slope,
- ρ: skewness parameter,
- m: horizontal shift,
- σ: curvature parameter.

This parametrization ensures a smooth fit across strikes and maturities.

9.2 SVI-JW (Jump-Wings Reparametrization)

Gatheral's SVI-JW form improves interpretability and control of the wings:

- Captures steep short-term skews due to jumps or leverage effects.
- Provides explicit control of left/right tail behavior, which is particularly relevant in cryptocurrencies with asymmetric risks.

9.3 Skew and Convexity

Skew and convexity are extracted directly from the implied variance curve:

$$\text{Skew} = \frac{\partial w}{\partial k}, \quad \text{Convexity} = \frac{\partial^2 w}{\partial k^2}$$

- **Skew:** captures the asymmetry between calls and puts, often linked to downside risk aversion.
- **Convexity:** measures curvature of the smile, indicating the strength of extreme-tail pricing.

9.4 Role in the Framework

In this project, IV smiles are not used to generate signals directly but serve as an **external cross-check**:

- Compare realized vs. implied volatility to compute the Variance Risk Premium (VRP).
- Validate whether the Markov-based probability distributions are consistent with option-implied risk perceptions.
- Provide an academic link between empirical volatility modeling and derivative pricing.

By embedding implied volatility analysis into the broader system, the framework gains an additional perspective on market expectations, strengthening both robustness and credibility.

Chapter 10

Fourier Methods for Options

Fourier-based pricing techniques provide efficient and theoretically elegant methods for valuing options under a wide range of models, particularly those with stochastic volatility or jumps. While not directly implemented in the Crypto Signal Generator, these methods form part of the broader mathematical toolbox considered in the project. Their relevance lies in providing alternative benchmarks and extending the system's theoretical foundations.

10.1 Carr–Madan FFT Approach

Carr and Madan (1999) introduced a method to compute European option prices using the Fast Fourier Transform (FFT). By exploiting the characteristic function $\phi(u)$ of the log-price under the risk-neutral measure, the call option price can be expressed as:

$$C(K) = e^{-\alpha k} \frac{1}{\pi} \int_0^\infty \Re \left(e^{-iuk} \frac{\phi(u - (\alpha + 1)i)}{\alpha^2 + \alpha - u^2 + i(2\alpha + 1)u} \right) du$$

where $k = \ln(K)$, $\alpha > 0$ is a damping factor, and K is the strike.

- Efficient computation through FFT makes this method attractive for calibrating models like Heston or Merton.
- Particularly relevant in crypto markets, where fat tails and jumps require non-Black–Scholes models.

10.2 PROJ Method (Fang–Oosterlee)

The PROJ method uses cosine and polynomial expansions to approximate the density function:

- Approximates option prices by projecting the density onto a finite set of basis functions.
- Offers faster convergence than FFT in some settings.
- Especially effective for exotic or path-dependent derivatives.

10.3 Role in the Framework

Although Fourier methods are not part of the live Python implementation, their role in the framework is threefold:

1. **Theoretical Benchmark:** Provide a rigorous way to validate pricing under stochastic volatility and jump models.
2. **Academic Completeness:** Ensure the system is positioned within the broader landscape of computational finance.
3. **Future Extension:** Enable integration of options-based forecasts into the probabilistic signal generator.

By acknowledging Fourier methods, the framework situates itself at the intersection of practical data-driven modeling and advanced option pricing theory.

Chapter 11

Validation and Metrics

Validation is a critical component of the Crypto Signal Generator. Since the framework produces probabilistic forecasts rather than deterministic predictions, the evaluation must focus not only on trading performance but also on the **statistical quality of probability estimates**. To this end, the system employs both **probabilistic scoring rules** and **portfolio-level performance metrics**.

11.1 Brier Score

The Brier score evaluates the accuracy of probabilistic forecasts against binary outcomes:

$$\text{Brier} = \frac{1}{N} \sum_{t=1}^{N} (p_t - y_t)^2$$

where p_t is the forecast probability of an upward move, and $y_t \in \{0, 1\}$ is the realized outcome.

- Lower values indicate better probabilistic calibration.
- Sensitive to both discrimination and reliability.

11.2 Log-Loss (Cross-Entropy)

Log-loss penalizes overconfident and incorrect forecasts more heavily:

$$-\frac{1}{N} \sum_{t=1}^{N} \left[y_t \log p_t + (1 - y_t) \log(1 - p_t) \right]$$

- Encourages well-calibrated probabilities.
- Particularly suited for imbalanced datasets, as often seen in directional crypto markets.

11.3 Sharpe Ratio

At the portfolio level, profitability is measured using the Sharpe ratio:

$$\text{Sharpe} = \frac{\mathbb{E}[R]}{\sigma_R}$$

where $\mathbb{E}[R]$ is the expected excess return, and σ_R is the standard deviation of returns.

- Captures the trade-off between reward and risk.
- Higher Sharpe ratios indicate more efficient risk-adjusted performance.

11.4 Maximum Drawdown

Risk control also requires evaluating downside exposure. Drawdown at time t is defined as:

$$DD_t = 1 - \frac{V_t}{\max_{s \leq t} V_s}$$

where V_t is the portfolio value.

- Provides insight into worst-case capital erosion.
- Essential for practical deployment in volatile crypto markets.

11.5 Combined Validation Framework

- **Probabilistic Scores (Brier, Log-loss):** Assess whether forecasts are statistically sound.
- **Portfolio Metrics (Sharpe, Drawdown):** Evaluate whether signals are economically viable.
- **Complementarity:** Both perspectives are required to ensure that probability forecasts translate into actionable and profitable strategies.

This dual validation framework ensures the system is not only **statistically calibrated** but also **financially meaningful**, bridging the gap between academic rigor and trading relevance.

Chapter 12

Results

The Crypto Signal Generator was tested on both historical and real-time data from Binance, focusing on BTC/USDT and extending to multiple cryptocurrency pairs. The results highlight the system's ability to produce stable probabilistic forecasts, generate interpretable fan charts, and deliver actionable trading signals under risk-aware constraints.

12.1 Probability Stability and State Transitions

- The Markov chain core consistently produced smooth probability distributions over market regimes, avoiding excessive volatility in the forecasts.
- Regime persistence was observed: once the system classified the market as being in a "mild uptrend" or "neutral" state, probabilities tended to remain stable unless strong shocks occurred.
- Entropy diagnostics provided additional value: high-entropy forecasts (uncertain) were associated with low signal quality, prompting cautious decision-making.

12.2 Multi-Horizon Fan Charts

- Fan charts showed widening uncertainty bands as horizons increased, consistent with theoretical expectations.
- Short horizons (1m, 5m) produced narrow intervals, reflecting higher confidence in immediate predictions.
- Longer horizons (1h, 4h) produced wider uncertainty ranges, illustrating the growth of forecast variance.
- This enabled traders to align strategies with their preferred time horizon and risk tolerance.

12.3 Signal Generation and Filtering

- Probability thresholds and hysteresis filters reduced noise and prevented frequent signal flipping.

- In trending environments, signals aligned well with sustained moves, producing actionable entries.
- Cost-aware thresholds filtered out trades with insufficient edge, accounting for fees and slippage.

12.4 Position Sizing and Risk Control

- The trimmed Kelly sizing method dynamically adjusted exposure based on probability edge and volatility conditions.
- During high-volatility periods, position sizes decreased, limiting downside risk.
- In stable conditions with strong signals, position sizes increased, enhancing capital efficiency.

12.5 Validation Metrics

- Brier scores and log-loss values confirmed the statistical calibration of probabilistic forecasts.
- Positive Sharpe ratios were observed across different configurations, validating profitability.
- Maximum drawdowns remained within acceptable levels when hysteresis and Kelly trimming were applied.

12.6 Performance Summary

The overall performance can be summarized as follows:

1. Probabilistic forecasts were stable and interpretable.
2. Multi-horizon analysis provided additional robustness.
3. Cost-aware and hysteresis filters reduced noise and overtrading.
4. Position sizing improved risk-adjusted returns while maintaining controlled drawdowns.

These results demonstrate that the system balances academic rigor with practical applicability, achieving both statistical soundness and economic value.

Chapter 13

Discussion

The Crypto Signal Generator demonstrates the potential of probabilistic and risk-aware modeling in cryptocurrency markets. By combining volatility-based features, Markov dynamics, and structured risk management rules, the framework bridges the gap between academic finance and practical trading.

13.1 Strengths and Contributions

1. **Interpretability and Transparency:** Each module corresponds to a well-defined mathematical object, ensuring that forecasts are not "black box" outputs but interpretable probability distributions.

2. **Multi-Horizon Perspective:** Propagation of transition matrices enables scenario analysis across different holding periods. Fan charts provide a visual and probabilistic representation of uncertainty.

3. **Risk-Aware Design:** Cost-aware thresholds and hysteresis filters reduce noise, while trimmed Kelly position sizing ensures capital is allocated dynamically and responsibly.

4. **Multi-Asset Scalability:** The modular architecture allows the framework to analyze multiple cryptocurrencies consistently, extending beyond BTC/USDT to other trading pairs.

5. **Academic Integration:** The inclusion of volatility models (Heston, Merton, Bates) and implied volatility parametrizations (SVI, SVI-JW) ensures the system remains connected to established financial theory.

13.2 Limitations and Practical Considerations

1. **Data Dependency:** The framework depends on continuous, high-quality Binance data. Latency or outages could compromise live forecasts.

2. **Discretization of States:** Market regimes are simplified into discrete categories, which, while interpretable, may omit continuous nuances.

3. **Model Assumptions:** Conditional Markov chains assume transitions depend on selected features. More complex nonlinear dynamics may remain unmodeled.

4. **Computational Constraints:** Multi-horizon probability propagation and feature extraction can be computationally intensive, especially in multi-asset real-time scenarios.

5. **Deployment Risks:** Live trading introduces factors such as liquidity constraints, slippage, and execution delays that may reduce real-world performance relative to backtests.

13.3 Implications

The system highlights how **probabilistic forecasting and structured risk management** outperform naive trading heuristics. It demonstrates that academic rigor (Markov chains, volatility modeling, implied volatility) can be integrated into a practical system. At the same time, it acknowledges the importance of adaptivity and operational constraints in live market deployment.

Overall, the Crypto Signal Generator provides a blueprint for combining theoretical finance with applied trading, showcasing how uncertainty can be quantified and harnessed for decision-making.

Chapter 14

Conclusion

The Crypto Signal Generator represents a comprehensive and finalized framework for probabilistic forecasting and risk-aware signal generation in cryptocurrency markets. Implemented in Python and deployed in practice, the system operates as a **multi-asset platform**, processing data from multiple cryptocurrencies and applying a consistent modular architecture across all assets.

The framework combines volatility-based feature engineering, Markov chain modeling, and multi-horizon forecasting with a structured decision layer. Forecasts are transformed into actionable strategies through cost-aware thresholds, hysteresis filters, and trimmed Kelly position sizing. This ensures that signals are not only statistically grounded but also economically viable when deployed in live trading environments.

Validation through proper scoring rules (Brier score, log-loss) confirms that forecasts are well calibrated, while portfolio-level metrics (Sharpe ratio, drawdown) demonstrate risk-adjusted profitability. The results highlight the system's ability to balance **academic rigor with operational applicability**, making it suitable for real-world use.

While the system is fully functional and already in the market, future enhancements could focus on integrating **machine learning models trained on backtesting data**. Such extensions could refine probabilistic forecasts, improve adaptability, and enhance signal quality by learning nonlinear dependencies not captured by the Markov framework.

In conclusion, the Crypto Signal Generator stands as a finished, market-ready product. It demonstrates how probabilistic reasoning, volatility analysis, and structured risk management can be applied effectively in volatile cryptocurrency markets, bridging the gap between theoretical finance and practical execution.

Chapter 15

References / Appendix

15.1 Mathematical Appendix

15.1.1 Returns and Volatility

$$r_t = \ln \frac{p_t}{p_{t-1}}$$

$$\sigma^2_{RV,t} = \frac{1}{N-1} \sum_{j=1}^{N} (r_{t-j} - \bar{r})^2$$

$$\sigma^2_t = (1-\lambda) r^2_t + \lambda \sigma^2_{t-1}, \quad \lambda \in [0.9, 0.99]$$

$$ATR_t = \frac{1}{n} \sum_{i=1}^{n} \max\{H_i - L_i, |H_i - C_{i-1}|, |L_i - C_{i-1}|\}$$

$$u_t = \frac{r_t}{\sigma_t + \varepsilon}$$

15.1.2 Markov Chains

$$P_{ij} = \Pr(S_{t+1} = j \mid S_t = i)$$

$$\widehat{P}_{ij} = \frac{N_{ij} + \alpha}{\sum_j N_{ij} + K\alpha}$$

$$\pi_{t+h} = \pi_t P^h$$

$$\pi^* = \pi^* P, \quad \sum_i \pi^*_i = 1$$

$$\Pr(S_{t+1} = j \mid S_t = i, x_t) = \frac{e^{\beta_{ij}^\top x_t}}{\sum_\ell e^{\beta_{i\ell}^\top x_t}}$$

$$H(\pi_{t+1}) = -\sum_j \pi_{t+1,j} \ln \pi_{t+1,j}$$

15.1.3 Multi-Horizon Forecasting

$$\mathbb{E}[R_H] = \sum_{h=1}^{H} \sum_{k} \pi_{h,k}\, \mu_k\, \sigma_t$$

$$\mathbb{V}[R_H] \approx \sum_{h=1}^{H} \sum_{k} \pi_{h,k}\, \upsilon_k\, \sigma_t^2$$

$$\hat{P}^q_{t+h} = P_t \exp\Big(\mathbb{E}[R_h] \pm z_q \sqrt{\mathbb{V}[R_h]}\Big)$$

15.1.4 Signals and Risk

$$p_t^{UP} = \pi_{t+1,4} + \pi_{t+1,5}, \quad p_t^{DN} = \pi_{t+1,1} + \pi_{t+1,2}$$

$$\tau_t = \frac{c_{round}}{k\sigma_t} + \tau_0$$

$$\text{Long if } p^{UP} > \tau_t + \delta, \quad \text{Close if } p^{UP} < \tau_t - \delta$$

$$w_t = \text{clip}\left(\kappa \cdot \frac{p_t^{UP} - p_t^{DN}}{\tau_t} \cdot \frac{\sigma_{target}}{\sigma_t} \cdot \frac{1}{1+g_t},\ 0,\ w_{max}\right)$$

15.1.5 Volatility and Jump Models

$$C = S_0 N(d_1) - Ke^{-rT} N(d_2)$$

$$dS_t = \mu S_t dt + \sqrt{v_t} S_t dW_t^{(1)}$$

$$dv_t = \kappa(\theta - v_t)dt + \xi \sqrt{v_t} dW_t^{(2)}$$

$$\frac{dS_t}{S_t} = (\mu - \lambda k)dt + \sigma dW_t + dJ_t$$

$$J \sim \text{DE}(\eta_1, \eta_2)$$

$$dS_t = S_t(\mu dt + \sqrt{v_t} dW_t^{(1)} + dJ_t)$$

15.1.6 Implied Volatility and Smiles

$$w(k) = a + b\left(\rho(k-m) + \sqrt{(k-m)^2 + \sigma^2}\right)$$

$$\text{Skew} = \frac{\partial w}{\partial k}, \quad \text{Convexity} = \frac{\partial^2 w}{\partial k^2}$$

15.1.7 Fourier Methods

$$C(K) = e^{-\alpha k} \frac{1}{\pi} \int_0^\infty \Re\left(e^{-iuk} \frac{\phi(u - (\alpha+1)i)}{\alpha^2 + \alpha - u^2 + i(2\alpha+1)u}\right) du$$

15.1.8 Validation Metrics

$$\text{Brier} = \frac{1}{N}\sum_t (p_t - y_t)^2$$

$$-\frac{1}{N}\sum_t \left[y_t \log p_t + (1-y_t)\log(1-p_t)\right]$$

$$\text{Sharpe} = \frac{\mathbb{E}[R]}{\sigma_R}$$

$$DD_t = 1 - \frac{V_t}{\max_{s\le t} V_s}$$

$$VRP_t = IV_t^2 - RV_t^2$$

15.2 References

- Binance API Documentation: `https://binance-docs.github.io/apidocs/`
- J.P. Morgan RiskMetrics Technical Document (1996).
- Hamilton, J.D. (1994). *Time Series Analysis.* Princeton University Press.
- Gatheral, J. (2006). *The Volatility Surface.* Wiley Finance.
- Carr, P., & Madan, D. (1999). "Option Valuation Using the Fast Fourier Transform." *Journal of Computational Finance.*
- Fang, F., & Oosterlee, C.W. (2008). "A Novel Pricing Method for European Options Based on Fourier-Cosine Series Expansions." *SIAM Journal on Scientific Computing.*
- Kelly, J.L. (1956). "A New Interpretation of Information Rate." *Bell System Technical Journal.*