Pipelin Architecture Document

Bitcoin Volatility Prediction Pipeline

1. Purpose of the Pipeline Architecture

The pipeline architecture explains the end-to-end flow of data in the system—from raw dataset ingestion to final volatility prediction. This document focuses on how data moves, not on implementation details or code.

The pipeline ensures:

- Correct time-series handling
- No data leakage
- Logical sequencing of preprocessing, modeling, and evaluation

2. High-Level Pipeline Overview

The system follows a sequential time-series machine learning pipeline consisting of the following stages:

- Data Ingestion
- Data Filtering

1. Data Cleaning & Preparation
2. Volatility Computation
3. Feature Engineering
4. Exploratory Data Analysis (EDA)
5. Model Training & Testing
6. Model Evaluation
7. Result Visualization

Each stage receives processed output from the previous stage and passes structured data to the next.

Step 1: Data Ingestion

- Historical cryptocurrency market data is loaded from a compressed CSV file.
- The dataset contains OHLC prices, volume, market capitalization, and cryptocurrency identifiers.
- Data is read into a structured DataFrame for processing.

Output: Raw cryptocurrency dataset

Step 2: Cryptocurrency Filtering

- The dataset is filtered to include Bitcoin only.
- This reduces complexity and ensures consistent volatility behavior.
- All further processing is performed exclusively on Bitcoin data.

Output: Bitcoin-specific historical dataset

Step 3: Data Cleaning and Preparation

This step ensures time-series reliability and data quality.

Actions performed:

- Removal of unnecessary index-like columns
- Conversion of date column into datetime format
- Sorting data in chronological order
- Removal of rows with zero trading volume
- Validation of missing values

Output: Clean, time-ordered Bitcoin dataset

Step 4: Volatility Computation (Core Pipeline Stage)

Volatility is not directly available and must be derived.

Pipeline logic:

- Daily log returns are calculated using consecutive closing prices
- A 7-day rolling standard deviation is computed to represent volatility
- The next-day volatility is defined as the prediction target
- This transforms price data into a supervised learning target.

Output: Dataset with volatility and target volatility

Step 5: Feature Engineering

To capture market dynamics, additional predictive features are generated:

- Absolute returns to measure price movement intensity
- High–Low price range to capture intraday variation
- Liquidity ratio using volume and market capitalization
- Lagged volatility features (1, 3, 7, 14 days)
- Rows containing undefined values due to rolling calculations are removed.

Output: Feature-enriched dataset ready for modeling

Step 6: Exploratory Data Analysis (EDA)

EDA is integrated into the pipeline to validate assumptions.

Performed analyses:

- Bitcoin closing price trend visualization
- Volatility trend over time
- Correlation heatmap of numerical features
- EDA confirms the relevance of engineered features and temporal patterns.

Output: Analytical understanding of feature relationships

Step 7: Data Preparation for Modeling

- Selected features are separated into input variables (X)
- Target volatility is assigned as output variable (Y)
- A time-based train-test split (80% train, 20% test) is applied
- This preserves chronological order and prevents future data leakage.

Output: Training and testing datasets

Step 8: Model Training

A Random Forest Regressor is trained on historical data

The model learns non-linear relationships between lagged volatility, price movement, and future volatility

Hyperparameters are selected to control overfitting and improve generalization

Output: Trained volatility prediction model

Step 9: Model Evaluation

- Model predictions on test data are evaluated using:
- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- R^2 Score

Comparison with a naïve baseline (previous-day volatility)

- This validates the effectiveness of the machine learning pipeline.

Output: Quantitative performance metrics

Step 10: Visualization and Interpretation

- Final pipeline outputs are visualized through:
- Feature importance plots
- Actual vs predicted volatility curves
- These visualizations support interpretability and result communication.

Output: Interpretable prediction results

4. Pipeline Characteristics

- Fully sequential and deterministic
- Time-series safe (no shuffling)
- Modular and extendable
- Suitable for other cryptocurrencies with minimal changes

5. Conclusion

The pipeline architecture provides a clear and structured data flow for Bitcoin volatility prediction. Each stage contributes logically to transforming raw market data into accurate next-day volatility predictions while maintaining data integrity and modeling reliability.