Crypto Volatility Detection

Real-Time AI Service for BTC-USD Spike Prediction

Presented by: Team Lead

Project Overview

- Objective: Predict short-term volatility spikes in BTC-USD markets
  - Binary classification: Spike (1) vs Normal (0)
  - 60-second lookahead prediction window
- Real-time streaming from Coinbase WebSocket API
  - Process 1+ tick/second with sub-800ms latency
- Full MLOps pipeline with monitoring and rollback
  - Docker-based deployment with one-command startup
  - Grafana dashboards for real-time observability

System Architecture

Data Flow

- Coinbase WebSocket → Ingestor
- Ingestor → Kafka (KRaft mode)
- Kafka → Feature Engineering
- Features → FastAPI /predict
- API → Prometheus metrics
- Prometheus → Grafana dashboards

Tech Stack

- API: FastAPI (Python 3.11)
- Streaming: Apache Kafka
- ML: scikit-learn (Logistic Regression)
- Tracking: MLflow
- Monitoring: Prometheus + Grafana
- Container: Docker Compose

Model Selection & Performance

Models Evaluated

- Z-Score Baseline: PR-AUC 0.33
- Random Forest: PR-AUC 0.30
- Gradient Boosting: PR-AUC 0.84
- Logistic Regression: PR-AUC 0.89 ✓
- Selected: Logistic Regression
- GridSearchCV hyperparameter tuning

Best Model Metrics

- PR-AUC: 0.8917
- ROC-AUC: 0.9399
- F1 Score: 0.9091
- Precision: 94.34%
- Recall: 87.72%
- Inference: 0.003 ms/sample

Feature Engineering & Training

Top Features (by importance)

- realized_volatility_300s (34%)
- realized_volatility_60s (29%)
- log_return_300s (16%)
- spread_mean_300s (9%)
- trade_intensity_300s (7%)
- order_book_imbalance (5%)

Training Details

- Dataset: 1,140 samples
- Train/Test Split: 80/20
- Class Balance: 57% normal, 43% spike
- 5-Fold Cross Validation
- GridSearchCV optimization
- All CPU cores utilized (n_jobs=-1)

API & Infrastructure

FastAPI Endpoints

- POST /predict - Volatility prediction
- GET /health - Service health check
- GET /version - Model version info
- GET /metrics - Prometheus metrics
- Rate limiting: 100 req/min
- Structured JSON logging

Infrastructure

- Docker Compose orchestration
- Kafka KRaft (no Zookeeper)
- Prometheus metrics collection
- Grafana dashboards
- CI/CD: GitHub Actions
- Black + Ruff linting

Backend/DevOps

Monitoring & SLOs

≤800ms p95 Latency Target	**99.5%** Availability	**<1%** Error Rate	**≥99%** Success Rate
Real-time CPU Usage	**Real-time** Memory	**Evidently** Drift Detection	**ml|baseline** Rollback

Backend/DevOps

Demo & Deliverables

- One-command startup: docker compose up -d
- API Contract: POST /predict with {rows: [...]}
  - Returns: {scores, model_variant, version, ts}
- Documentation suite:
  - team_charter.md - Roles & responsibilities
  - selection_rationale.md - Model choice
  - slo.md - Service Level Objectives
  - runbook.md - Operations guide
- Demo video: 8-minute walkthrough
  - Startup → Prediction → Failure Recovery → Rollback

All Team Members