AINewsQuake

Price Chart with News Annotations

Data Management Project Report

Course: Data Management and Data Visualization
Institution: University of Milano-Bicocca (UNIMIB)
Academic Year: 2025–2026
Repository: github.com/smabbasht/ainewsquake

Group Members:
Abbas Taqvi
Umeir Mohamed
MohammadAmin Saberi

January 10, 2026

Abstract

AINewsQuake is an event-driven data pipeline that investigates the relationship between *AI-related news sentiment* and *intraday stock price dynamics*. The system integrates heterogeneous data sources—a news feed enriched with sentiment scores and high-frequency (1-minute) OHLCV market data—and stores them in a time-series optimized database (TimescaleDB). An interactive Streamlit dashboard visualizes candlestick charts annotated with sentiment-coded news markers, enabling exploratory analysis of how news events align with price movements and volatility. The project demonstrates a complete data management workflow: acquisition, storage, profiling, integration, analysis, and quality improvement, with strong emphasis on reproducibility and scalable time-series querying.

Contents

1 Introduction

Financial markets react quickly to information. In the context of fast-moving technological trends, AI-related announcements and headlines can trigger substantial changes in market behavior. **AINewsQuake** was designed to quantify and visualize the "seismic impact" of AI news on price movements, capturing and integrating both **slow data** (text-based news events) and **fast data** (high-frequency market microstructure).

The project objective is twofold:

O1: Build a reproducible pipeline that acquires, stores, integrates, and validates heterogeneous data sources.

O2: Enable research-driven exploration of market reactions through SQL queries and an interactive dashboard.

1.1 Research Questions

The project answers the following research questions (RQs):

- **RQ1:** How does AI-related news sentiment correlate with intraday volatility and price movement for AI-centric stocks?
- **RQ2:** Do certain stocks react more strongly to positive or negative AI-related headlines (sentiment polarity)?
- **RQ3:** How complete and reliable is the integrated dataset, and what are the main integration and data quality issues?

1.2 Scope

We focus on 10 AI-centric tickers (NVDA, TSLA, MSFT, GOOGL, AAPL, AMD, PLTR, TSM, SMCI, META) across the year 2025. The system is built to be extendable to additional companies and time periods.

2 Data Sources and Acquisition

2.1 News Data (Finnhub API)

The project uses the Finnhub API for historical company news. Finnhub was selected due to its historical news availability and manageable rate limits (up to 60 calls/minute on the free tier). The API provides event metadata including publication timestamps, headline text, and source information.

2.2 Market Data (Databento API)

High-frequency market data (1-minute OHLCV) is obtained via the Databento API. This source was preferred over alternatives such as Yahoo Finance due to higher reliability for intraday historical bars and consistent timestamp handling.

2.3 Acquisition Constraints

Two practical acquisition constraints shape the design:

1. **API rate limits and pagination:** Finnhub returns at most 250 news items per request.
2. **Data volume:** 1-minute bars across many tickers and a full year produce large datasets requiring efficient storage and indexing.

3 Storage Layer and Database Design

3.1 Why TimescaleDB?

TimescaleDB is a PostgreSQL extension optimized for time-series workloads. It provides hypertables, time-based partitioning, native compression policies, and fast range queries. These features are particularly well-suited for market data and time-indexed news events.

3.2 Logical Schema

The pipeline stores two main entities:

- **News events table (`ai_news_events`):** stores article headline, timestamp, ticker, source, and sentiment score.
- **Market hypertable (`market_ticks`):** stores 1-minute OHLCV bars keyed by `(time, ticker)`.

3.3 Physical Design (Hypertables & Compression)

Market ticks are stored as a TimescaleDB hypertable partitioned by week. Compression is enabled for data older than 7 days, balancing storage efficiency with query performance.

4 Data Profiling and Data Quality

4.1 Validation and Type Safety

Before inserting data into the database, records are validated using Pydantic schemas, ensuring type correctness and preventing malformed values from entering persistent storage.

4.2 Idempotent ETL and De-duplication

The ETL pipeline is designed to be idempotent:

- **News events** are inserted with `ON CONFLICT DO NOTHING` using a unique event identifier.
- **Market ticks** are inserted with `ON CONFLICT DO UPDATE`, supporting safe re-runs and partial backfills.

This design prevents duplicate rows and ensures consistency across repeated runs.

4.3 Quality Metrics (Recommended Reporting)

For the final delivery, we recommend reporting key data quality metrics:

- **Completeness:** missing values in key fields (timestamp, headline, OHLCV).
- **Uniqueness:** duplicate event IDs and duplicate (time, ticker) pairs.
- **Validity:** timestamp format correctness; sentiment score range $[-1, 1]$; non-negative volume.
- **Consistency:** timezone normalization and alignment between news timestamps and market bars.

5 Data Integration Strategy

5.1 Heterogeneous Integration Challenge

The core integration challenge is to align:

- **Event-time data (news)** — irregular timestamps.
- **Time-series bars (OHLCV)** — uniform 1-minute timestamps.

This requires temporal alignment and careful handling of time zones.

5.2 Join-at-Query-Time

Instead of materializing a single massive joined table, **AINewsQuake** keeps sources separate and performs integration at query time for visualization and analytics. This design:

- preserves raw sources,
- enables flexible alignment windows (e.g., nearest-minute bar),
- reduces duplication and storage overhead.

5.3 Integration Error Measurement

Integration quality can be measured by:

- **Coverage:** fraction of news events that can be mapped to at least one market bar.
- **Latency:** time difference between news timestamp and nearest candle timestamp.
- **Ambiguity:** cases where multiple news map to the same candle.

6 Analysis and Visualization

6.1 Dashboard Overview

The Streamlit dashboard provides:

- interactive candlestick charts,
- sentiment-coded news markers,
- hover tooltips showing headline and sentiment,
- total volume visualization and sentiment summaries,
- ticker selection and date-range filters.

6.2 Suggested SQL Queries

To support the RQs, the following example queries can be executed on TimescaleDB:

- **News volume and sentiment by ticker:** count events and average sentiment.
- **Volatility around news windows:** compute volatility for $t \pm k$ minutes around events.
- **Outlier detection:** detect extreme price moves and inspect associated headlines.

7 Reproducibility and Deployment

The project is fully reproducible using Docker and environment variables:

1. Create a `.env` file containing API keys for Finnhub and Databento.
2. Start TimescaleDB with Docker Compose.
3. Initialize the database schema.
4. Run ETL for news ingestion and market ingestion.
5. Launch the Streamlit application at `localhost:8501`.

7.1 Operational Notes

The repository uses `uv` for Python dependency management. The modular repository-service structure enables maintainability and future extensions (additional tickers, additional data sources, alternative sentiment models).

8 Limitations and Future Improvements

8.1 Limitations

- Free-tier API limits may cause incomplete backfills if not carefully scheduled.
- Sentiment scoring uses VADER; domain-specific models could provide improved accuracy.
- Correlation does not imply causation; stronger causal inference methods could be explored.

8.2 Future Improvements

- Add more advanced integration windows (e.g., pre/post-event windows).
- Implement anomaly detection models to identify "quake" events systematically.
- Add automated data quality dashboards (metrics over time).
- Support more markets (crypto, forex) with additional APIs.

9 Conclusion

AINewsQuake demonstrates an end-to-end data management pipeline integrating text-based news with high-frequency market data. The project emphasizes robust acquisition (API backfill under constraints), efficient time-series storage (hypertables and compression), data validation, idempotent loading, and scalable integration for analysis. The Streamlit dashboard enables intuitive exploration of AI news impact on market behavior, addressing the core research questions and satisfying the required phases of the course project.

References

- Project repository: github.com/smabbasht/ainewsquake
- TimescaleDB documentation: https://docs.timescale.com/
- Finnhub API documentation: https://finnhub.io/docs/api
- Databento documentation: https://databento.com/
- VADER sentiment: https://github.com/cjhutto/vaderSentiment