Project: Autonomous AI-Driven Signal Discovery Framework

Overview

This project defines a modular, extensible architecture for a closed-loop financial signal discovery platform that integrates: - Self-supervised and symbolic feature learning - GPT-5 agentic reasoning for planning and retraining - Risk-aware backtesting and strategy evaluation - Vector-based signal memory and similarity search

It is designed to support any time-series-driven use case, including but not limited to: - Insider trading pattern detection - Systematic anomaly detection - Market regime classification - Macro event signal generation - Execution alpha optimization

Technical Architecture Stack

🧱Infrastructure Layer

Component	Technology
Compute orchestration	Docker + Kubernetes (or local dev with `task` / `Makefile`)
Workflow pipelines	Airflow / Prefect / Dagster
Storage	PostgreSQL + `pgvector` + S3 / GCS / Parquet on disk
Vector DB	FAISS (local), Pinecone / Weaviate (managed), Milvus (on-prem)
Logging	Weights & Biases, MLflow, Supabase tables, or OpenTelemetry

🧰Modular Subsystems

1. Data Ingestion + Preprocessing

- Source data: price/volume, LOB, SEC/EDGAR, dark pool (ATS/TRF), options, news
- Tools: `pandas`, `pydantic`, `polars`, `lxml`, `sec-edgar-downloader`
- Output: Time-indexed Parquet datasets (per asset, time window)

2. Feature Generator Stack

- **Self-supervised encoders**: `ts2vec`, `series2vec`, `DeepLOB`, `InceptionTime`
- **Symbolic regression**: `pysr`, `AI-Feynman`, symbolic distillation from NN layers
- **TDA / Topology**: `giotto-tda`, `ripser`, `scikit-tda`
- **Microstructure + Anomaly**: Custom logic for OFI, spread z-scores, bid-ask imbalance
- **Change-point detection**: `ruptures`, `bocpd`, `hmmlearn`, `tick.hawkes`

- Output: Tabular and vector representations of signals per window

3. Vector Storage + Retrieval

- Feature vectors indexed using FAISS / Pinecone
- Metadata: timestamps, asset, market regime, labeling info
- Supports: KNN queries, regime-filtered search, hybrid search (symbol + vector)

4. Modeling + Strategy Evaluation

- ML: `xgboost`, `lightgbm`, `catboost`, `sklearn`, `pytorch`
- Backtesting: `vectorbt`, `bt`, `zipline-reloaded`, custom PnL engine
- Labeling: Supervised (known outcome), weakly supervised (event-horizon triggers)
- Metrics: Sharpe, Sortino, drawdown, hit rate, regime stability

5. GPT Agentic Layer

- Tools: `LangGraph`, `OpenAgents`, `AutoGen`, `CrewAI`
- Accesses:
- Backtest logs / signal metadata
- Feature pipeline codebase
- Promptable API for: `retrain_model()`, `add_feature()`, `drop_feature()`
- Capabilities:
- Planning retrains, proposing new feature chains
- Debugging strategy decay / drift
- Symbolic summarization of discovered features

6. Continual Learning Loop

- Rolling retrain pipelines
- Feature pruning + survival tracking
- Embedding reindexing and vector drift handling
- Prompt-based agent interventions (e.g. "Explain why this feature failed last week")

System Use Case Plugability

This architecture is **fully modular and use-case agnostic**. Each use case is simply a: - **New anchor event definition** (e.g. Form 4 date, macro release, price spike) - **Custom labeling rule** (e.g. did insider file X days later, did volatility spike Y days later) - **Specialized feature subset** (e.g. add news embeddings or TDA for one strategy)

The rest of the pipeline remains unchanged: the agent, feedback loop, retraining, and vector storage adapt automatically. This enables: - Parallel use cases coexisting (with independent vector indexes) - Regime-aware retrieval across domains - Scaling to any time-series or event-based alpha problem

Next Steps

- Define first use case (e.g. insider Form 4 anchored alpha)

- Set up GitHub repo with modular folders: ingestion, features, models, db, agent
- Run agentic prompt loop over backtest logs
- Deploy vector DB with embedded signals
- Evaluate retrieval quality and strategy recall accuracy

This stack future-proofs the system for continual AI-driven financial intelligence evolution.