Factor Extraction from Macroeconomic News Streams to Drive Agentic Financial Trading Strategies

Chuan Bin Phoe, Neaton Ang

Problem Statement

Motivation

- Traditional time-series models tend to ignore qualitative macro and industry narratives
- Financial news is noisy, unstructured, and difficult to translate into actionable signals
- We need a structured pipeline that extracts relevant news, summarizes key points, and generates interpretable sentiment for forecasting

Training - Data Loader

OOP: TrainDataLoader class

```
config = Config("config.env")
print(f"Config: {config}")
train_data_loader = TrainDataLoader(config)
```

Download Bloomberg News data using HF datasets

Hugging Face

Bloomberg financial news dataset

446k rows

```
Downloading dataset 'danidanou/Bloomberg_Financial_News' with split 'train' from the Hugging Face Hub...
bloomberg_financial_data.parquet.gzip: 100% 482M/482M [00:02<00:00, 430MB/s]
Generating train split: 100% 446762/446762 [00:08<00:00, 51255.07 examples/s]

--- Download Successful! ---
```

Data Validation using Pydantic Class

Pydantic

Pydantic objects

Ensure data integrity

```
--- Starting validation of 446762 entries ---
Validating entries: 100%|██████████| 446762/446762 [00:28<00:00, 15910.17it/s]
--- Validation Complete! ---
Training dataset validated.
Saving processed dataset to local cache at '../data/danidanou_Bloomberg_Financial_News_train'...
Total number of rows: 446762
Loading pipeline completed.
```

Parquet

Parquet is a binary, columnar storage format
Cloud ready: *Speed and Compression > CSV*

Data Caching in Parquet files

Subsequent requests for that data can be served more quickly without needing to retrieve it from the original source

Dataset	Size on Amazon S3	Query Run Time	Data Scanned	Cost
Data stored as CSV files	1 TB	236 seconds	1.15 TB	$5.75
Data stored in Apache Parquet Format	130 GB	6.78 seconds	2.51 GB	$0.01
Savings	87% less when using Parquet	34x faster	99% less data scanned	99.7% savings

Databricks https://www.databricks.com/glossary/what-is-parquet

Training - Data Processor

OOP: NewsProcessor class

```
processor = NewsProcessor(config)
```

Processing Pipeline

```python
from processor import NewsProcessor
import nest_asyncio
nest_asyncio.apply()

processor = NewsProcessor(config)
sample = processor.remove_redundant_info(train_ds[DATA_START:DATA_END])
df = processor.enrich_news_entries_with_classifications(sample, save_path=f"{GDRIVE_PATH}processed_news")
df = processor.group_by_date_and_industry(df, save_path=f"{GDRIVE_PATH}grouped_news")
df = processor.filter_and_analyze_news(df)
df = processor.extract_impactful_news(df, top_n=3, save_path=f"{GDRIVE_PATH}impact_news")
df = processor.get_consolidated_sentiment(df, save_path=f"{GDRIVE_PATH}sentiment_news")
final_df = await processor.get_explanation(df, save_path=f"{GDRIVE_PATH}explanation_news")
```

Auto-saving to Google Drive at every step

Headline	Link	Date	Article

1

Headline	Link	Date	Article	Industry	Sentiment

1

Load HF weights and run inference to get: **Hugging Face**

Sentiment	ProsusAI / FinBERT
Industry	MoritzLaurer / mDeBERTa-v3-base-mnli-xnli

```
FinBERT Sentiment: 100%|██████████| 1550/1550 [53:04<00:00,  2.05s/batch]
Industry Classification: 100%|██████████| 12399/12399 [6:27:58<00:00,  1.88s/batch]
Completed processing 396762 entries
```

Ran on Google Colab's Nvidia A100 GPU

2

Grouping all articles with same (Date, Industry) into same row
Filtering out Industry=None news, get analysis statistics

Training - Data Processor

OOP: NewsProcessor class

```
processor = NewsProcessor(config)
```

Processing Pipeline

```
from processor import NewsProcessor
import nest_asyncio
nest_asyncio.apply()

processor = NewsProcessor(config)
sample = processor.remove_redundant_info(train_ds[DATA_START:DATA_END])
df = processor.enrich_news_entries_with_classifications(sample, save_path=f"{GDRIVE_PATH}processed_news")
df = processor.group_by_date_and_industry(df, save_path=f"{GDRIVE_PATH}grouped_news")
df = processor.filter_and_analyze_news(df)
df = processor.extract_impactful_news(df, top_n=3, save_path=f"{GDRIVE_PATH}impact_news")
df = processor.get_consolidated_sentiment(df, save_path=f"{GDRIVE_PATH}sentiment_news")
final_df = await processor.get_explanation(df, save_path=f"{GDRIVE_PATH}explanation_news")
```

Auto-saving to Google Drive at every step

Headline	Link	Date	Article

1

Headline	Link	Date	Article	Industry	Sentiment

2

Date	Industry	ArticleCount	News

*Each row is 1 unique (Date, Industry) pair, **News** contains all articles for that pair*

1

Load HF weights and run inference to get: Hugging Face

Sentiment	ProsusAI / **FinBERT**
Industry	MoritzLaurer / **mDeBERTa**-v3-base-mnli-xnli

```
FinBERT Sentiment: 100%|██████████| 1550/1550 [53:04<00:00,  2.05s/batch]
Industry Classification: 100%|██████████| 12399/12399 [6:27:58<00:00,  1.88s/batch]
Completed processing 396762 entries
```

Ran on Google Colab's Nvidia A100 GPU

2

Grouping all articles with same (Date, Industry) into same row
Filtering out Industry=None news, get analysis statistics

```
================================================================
Dropped 1141 (Industry, Date) pairs with Industry='None'
Remaining pairs: 13591
================================================================

Summary Statistics:
Total unique (Industry, Date) pairs: 13591
Average articles per pair: 27.91
Max articles in a pair: 1262
Min articles in a pair: 1
25th percentile: 4.0
50th percentile: 13.0
75th percentile: 29.0
Number of pairs with at least 3 articles: 11284
Total articles: 379362
```

Training - Data Processor

OOP: NewsProcessor class

```
processor = NewsProcessor(config)
```

Processing Pipeline

```
from processor import NewsProcessor
import nest_asyncio
nest_asyncio.apply()

processor = NewsProcessor(config)
sample = processor.remove_redundant_info(train_ds[DATA_START:DATA_END])
df = processor.enrich_news_entries_with_classifications(sample, save_path=f"{GDRIVE_PATH}processed_news")
df = processor.group_by_date_and_industry(df, save_path=f"{GDRIVE_PATH}grouped_news")
df = processor.filter_and_analyze_news(df)
df = processor.extract_impactful_news(df, top_n=3, save_path=f"{GDRIVE_PATH}impact_news")
df = processor.get_consolidated_sentiment(df, save_path=f"{GDRIVE_PATH}sentiment_news")
final_df = await processor.get_explanation(df, save_path=f"{GDRIVE_PATH}explanation_news")
```

Auto-saving to Google Drive at every step

Date	Industry	ArticleCount	News

3

Date	Industry	ArticleCount	News	ImpactfulNews	SentimentScore

***Impactful News** contains 3 articles with the highest abs(sentiment score)*
***SentimentScore** is based on these 3 articles*

3

Get 3 most impactful news for each row (based on abs(sentiment score))
Rerun sentiment scoring with FinBERT based on them

- Market psychology: Big headlines move markets
- Cost constraints: Reduce input tokens for next step

```
Extracting top 3 impactful news per (Industry, Date) pair...
Processing groups: 100%|██████████| 13591/13591 [00:00<00:00, 15013.76it/s]
Processing 13591 news entries...
FinBERT Sentiment: 100%|██████████| 54/54 [01:46<00:00,  1.96s/batch]
Completed processing 13591 entries
```

4

Asynchronous LLM call to generate sentiment explanation

Based on:

- Impactful News
- FinBERT sentiment score
- General Market News from the **same day**

Output Validation using Pydantic Class

Training - Data Processor

OOP: NewsProcessor class

```
processor = NewsProcessor(config)
```

Processing Pipeline

```
from processor import NewsProcessor
import nest_asyncio
nest_asyncio.apply()

processor = NewsProcessor(config)
sample = processor.remove_redundant_info(train_ds[DATA_START:DATA_END])
df = processor.enrich_news_entries_with_classifications(sample, save_path=f"{GDRIVE_PATH}processed_news")
df = processor.group_by_date_and_industry(df, save_path=f"{GDRIVE_PATH}grouped_news")
df = processor.filter_and_analyze_news(df)
df = processor.extract_impactful_news(df, top_n=3, save_path=f"{GDRIVE_PATH}impact_news")
df = processor.get_consolidated_sentiment(df, save_path=f"{GDRIVE_PATH}sentiment_news")
final_df = await processor.get_explanation(df, save_path=f"{GDRIVE_PATH}explanation_news")
```

Auto-saving to Google Drive at every step

Date	Industry	ArticleCount	News	ImpactfulNews	SentimentScore

4

	Industry	Date	News	ArticleCount	ImpactfulNews	AvgSentimentScore	SentimentScore	SentimentExplanation
0	Communication Services	2011-10-06	[{'Headline': 'FCC to Revamp Phone Subsidy to ...	2	[{'Headline': 'Euro-Area Leaders to Hold Summi...	0.709191	-0.291917	Overall sentiment for the Communications Servi...
1	Consumer Discretionary	2011-10-06	[{'Headline': 'PepsiCo May Purchase Russian Dr...	1	[{'Headline': 'PepsiCo May Purchase Russian Dr...	0.881740	0.888237	The article's sentiment is strongly positive f...
2	Consumer Staples	2011-10-06	[{'Headline': 'Ukraine's Grain Harvest Advance...	1	[{'Headline': 'Ukraine's Grain Harvest Advance...	-0.918589	-0.917441	Explanation: FinBERT indicates a strongly nega...
3	Energy	2011-10-06	[{'Headline': 'Clean-Tech Companies Should Get...	9	[{'Headline': 'Norway Boosts Mongstad Carbon-S...	0.252093	-0.252850	The energy-angle sentiment is mildly negative,...

3

Get 3 most impactful news for each row (based on abs(sentiment score))
Rerun sentiment scoring with FinBERT based on them

- Market psychology: Big headlines move markets
- Cost constraints: Reduce input tokens for next step

```
Extracting top 3 impactful news per (Industry, Date) pair...
Processing groups: 100%|██████████| 13591/13591 [00:00<00:00, 15013.76it/s]
Processing 13591 news entries...
FinBERT Sentiment: 100%|██████████| 54/54 [01:46<00:00,  1.96s/batch]
Completed processing 13591 entries
```

4

Asynchronous LLM call to generate sentiment explanation

Based on:

- Impactful News
- FinBERT sentiment score
- General Market News from the **same day**

Output Validation using Pydantic Class

Forecasting Model

Modelling Objective.

Quantify whether structured news sentiment provides incremental predictive power beyond traditional market & price-based factors.

Data Overview.

- ~7 years of daily stock price data (AAPL, MSFT, AMZN, … + SPY)
- Daily returns, rolling volatilities, momentum, liquidity, etc.
- Industry assignment - (Agentic LLM Extracted)
- Key-point extraction - (Agentic LLM Extracted)
- Sentiment scoring (per-article + aggregated) - (Agentic LLM Extracted)

Experimental Setup

- **Data split: Walk-Forward split (Quantile at Date)**
  - Chronological split: the first 70% of dates are used for training and the remaining 30% of dates are used for testing
  - Model is trained only on past data and evaluated on a contiguous future block of data
  - No shuffling
  - This ensures that there is no leakage from the test period into training
- **Classification Metrics**
  - **Accuracy:** % of correct predictions; use when classes balanced and all errors equally costly
  - **AUC:** Ranks positives vs negatives across all thresholds; robust to class imbalance
- **Forecasting Metric**
  - **R-Square:** Fraction of target variance explained by predictions; standard for continuous-value forecasting (e.g., returns)

Experiment Setup: Baseline Clarification

- We use a “CAPM-style market-only baseline”, not the theoretical CAPM model
- It is simply a single-factor forecasting model using only forward SPY return
  - “Forward SPY return” = the cumulative percentage move of SPY from today to h days ahead
- This will show how much predictive structure exists without sentiment, factors, or non-linearities

Model V1: Trained on Indexes

We first train return-forecasting models on indexes.

SPY → market proxy

Sector ETFs (e.g., XLF, XLK) → Represents an industry

Using only.

- Market Factor (Forward SPY)
- Momentum/Volatility features
- Sentiment features

Purpose.

Understand baseline signal strength before going to noisier individual stocks

Results.

Market-only baseline (CAPM-style) wins index have betas close to 1 and highly correlated with SPY, so the CAPM market factor captures a large portion of their return variance.

Model V2: Trained on Individual Stocks

We then transition to individual stocks.

Harder problem → Lower autocorrelation in returns, sentiment may only matter on event-heavy days → but it is more realistic

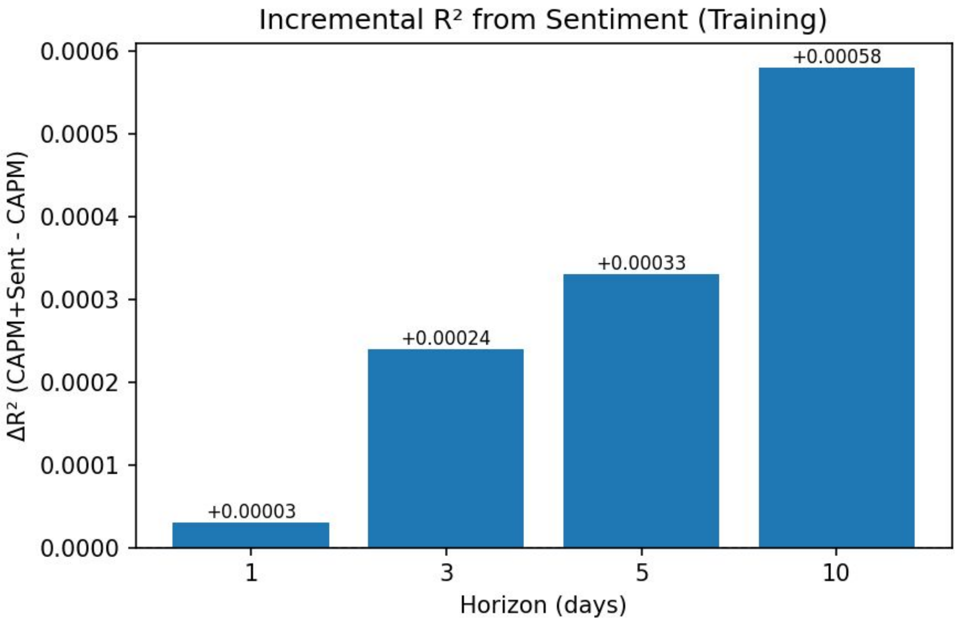
Key engineered features:

Category	Features
Price / Liquidity	ret, ret_3d, ret_5d, ret_vol_20d, ret_vol_60d, dollar_vol_rel_3d, dollar_vol_rel_20d
Sentiment	Standardized daily score, Rolling 3d/5d means & sums, Shock metric = (today – 5d mean)/std, News count z-score
Forward targets	ret_next_1d, ret_next_3d, ret_next_5d, ret_next_10d

Model V2: 4 Main Variants

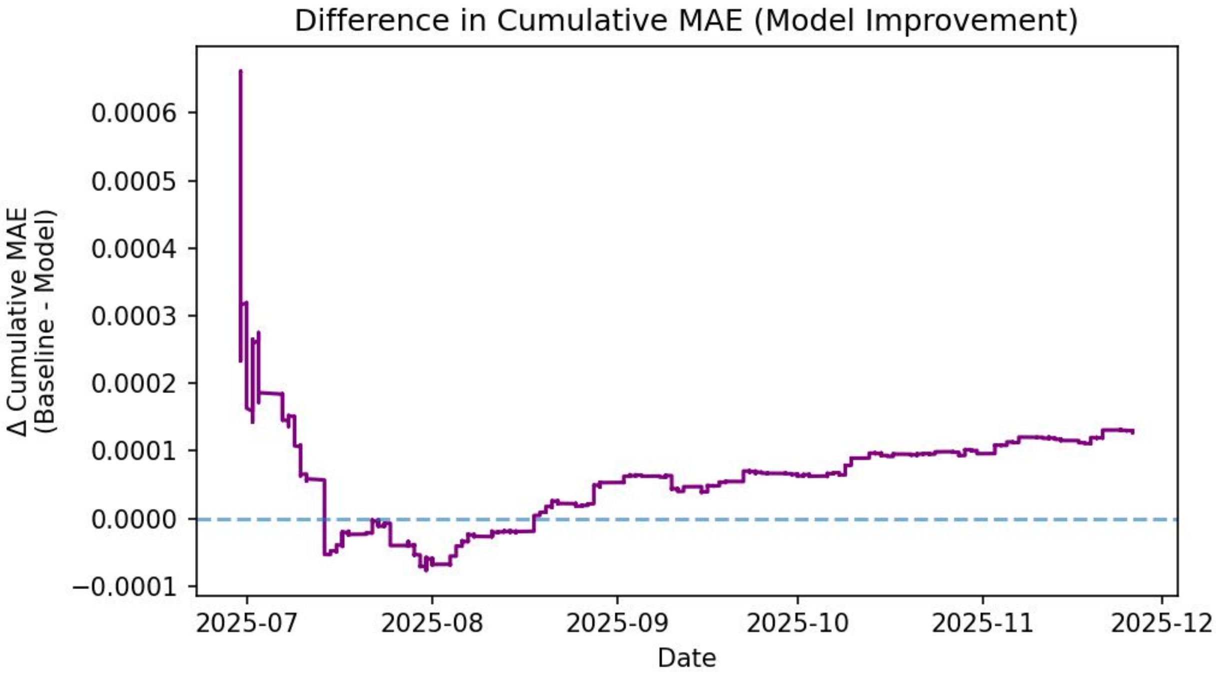
Model type	Predictors (per horizon h = 1D/3D/5D/10D)	Model family
CAPM-style baseline	MKT_h (forward SPY market factor)	LinearRegression / LogisticRegression
CAPM + Sentiment	CAPM baseline + sentiment history windows	LinearRegression / LogisticRegression
MF + Sentiment	MF + sentiment history windows	LinearRegression / LogisticRegression
Tree-based variants	MF + sentiment history windows	Gradient Boosting Regressor / Classifier + Random Forest

Model V2: Result Highlight (CAPM vs CAPM + Sentiment)



Sentiment adds a small but consistent directional improvement to CAPM

Model V2: Result Highlight (GB MF + Sentiment)

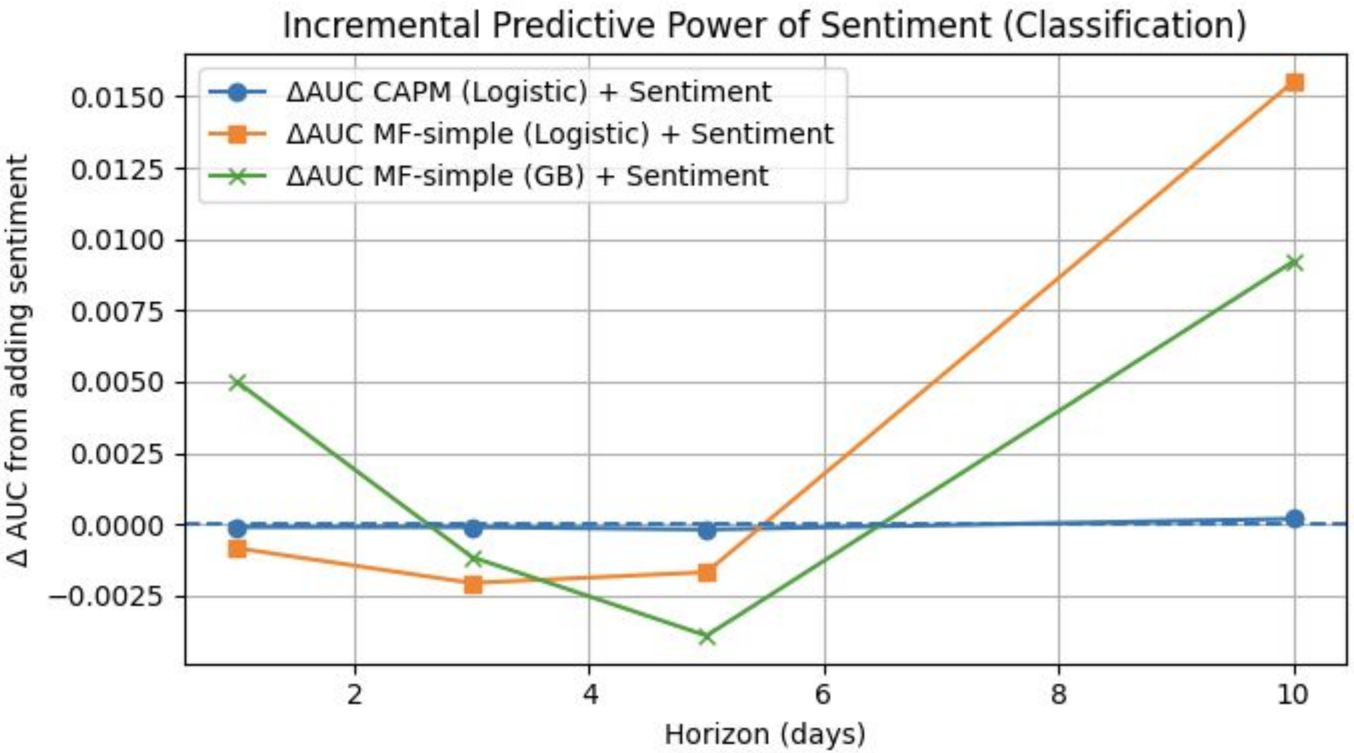


Model V2: Result Summary

Point	Summary	What it shows
1	Sentiment is a weak but consistently directional signal ($\Delta R^2 \approx 0.0002$–$0.0006$).	Even tiny R^2 gains matter in quant, where alpha is small.
2	Sentiment helps most on a simple CAPM-style baseline (market-only).	With only market exposure, sentiment adds extra information.
3	In multi-factor models, sentiment overlaps with price/volatility features.	Much news is already priced in, so sentiment sometimes adds noise.
4	Best performance occurs with GB + sentiment at 5D horizon.	At medium horizons, sentiment adds nonlinear predictive power.
5	Overall, news sentiment is weak but independent and horizon-dependent.	Short horizons are market-driven; longer horizons reflect sentiment and expectations.

Model V2: Classification Results

Classification collapses noise and focuses on direction, not magnitude



Model V2: Classification Results Findings

Horizon	What happens	Effect of sentiment
1–5 days (short)	Price/volume features dominate; microstructure noise is high.	Sentiment ≈ noise → small negative ΔAUC; little incremental value.
10 days (medium)	Short-term noise averages out; medium-term drift matters more.	Sentiment adds consistent value: +0.015 AUC (Logistic MF-simple), +0.009 AUC (GB MF-simple), small positive gain even on CAPM.
Overall takeaway	Predictive power of sentiment increases with horizon, consistent with findings that news tone predicts returns more at multi-day horizons than at 1-day horizons.	

Overall Findings

Theme	Summary	Takeaway
Weak but real signal	Sentiment adds small but consistent improvements (ΔR^2) on simple CAPM-style models.	Even tiny improvements matter in quant; the direction of effect is stable.
Horizon dependence	Impact grows at medium horizons (5–10 days).	Short-term noise dominates 1–3D; sentiment-driven drift shows up at longer horizons.
Independent information	Sentiment adds value beyond market, price, and liquidity features.	Helps CAPM models most; tree models capture extra nonlinear effects from sentiment.
Best use case	Sentiment performs best as an enhancer, not a standalone predictor.	Most useful in classification and tail-event detection, not raw return regression.
Overall verdict	**Sentiment is weak but horizon-dependent and consistently useful.** **Finer sentiment representation could provide a stronger signal.** **It provides incremental predictive power (ΔR^2, ΔAUC) when incorporated properly.**	

Modelling Future Work & Limitations

- We are using industry-aggregated sentiment → signal dilution could be high
- Sentiment representation here could still be a bit coarse
- Filter stock-level entity-matched sentiment
- Separate sentiment by type: earnings vs macro news, firm-specific vs sector vs market (Finer sentiment)
- Higher-frequency alignment → Incorporate intraday data with sentiment
- Study how quickly sentiment decays intraday and which events have multi-day follow through
- Feed classification outputs into regression as features

Inference using Agent - Tactical

Data Loader

- **Retrieve data from Bloomberg RSS Feeds**
- **Data Validation with Pydantic**

Data Processor

- **Same pipeline as Train**

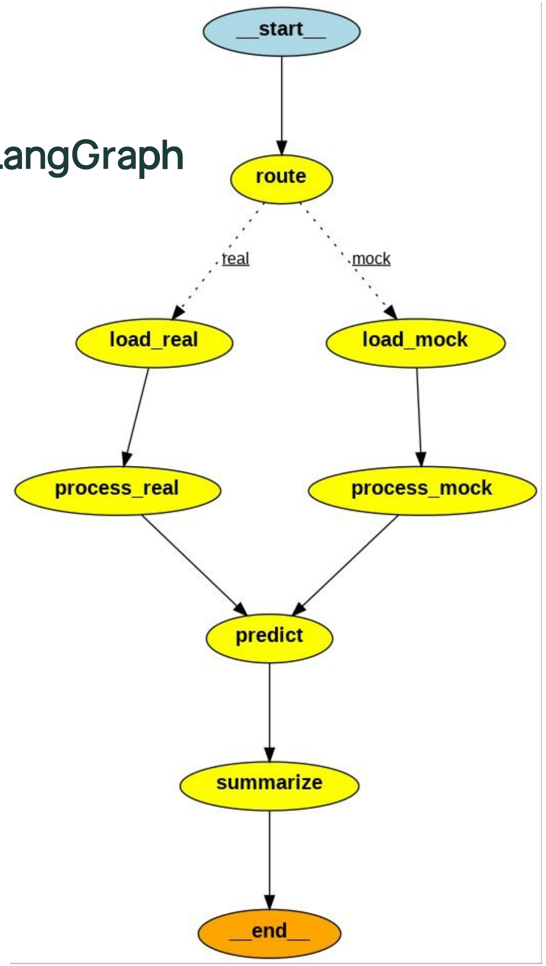
Model Prediction

- **Load Trained Model and perform Inference**

```python
feeds = [
    "https://feeds.bloomberg.com/news/news.rss",
    "https://feeds.bloomberg.com/markets/news.rss",
    "https://feeds.bloomberg.com/business/news.rss",
    "https://feeds.bloomberg.com/technology/news.rss",
    "https://feeds.bloomberg.com/politics/news.rss",
    "https://feeds.bloomberg.com/wealth/news.rss",
    "https://feeds.bloomberg.com/economics/news.rss",
    "https://feeds.bloomberg.com/green/news.rss",
    "https://feeds.bloomberg.com/pursuits/news.rss",
    "https://feeds.bloomberg.com/opinion/news.rss",
    "https://feeds.bloomberg.com/finance/news.rss",
    "https://feeds.bloomberg.com/real-estate/news.rss",
    "https://feeds.bloomberg.com/deals/news.rss",
    "https://feeds.bloomberg.com/crypto/news.rss",
    "https://feeds.bloomberg.com/europe/news.rss",
    "https://feeds.bloomberg.com/uk/news.rss",
    "https://feeds.bloomberg.com/asia/news.rss",
    "https://feeds.bloomberg.com/commodities/news.rss",
    "https://feeds.bloomberg.com/currencies/news.rss",
    "https://feeds.bloomberg.com/fixed-income/news.rss",
    "https://feeds.bloomberg.com/equities/news.rss",
    "https://feeds.bloomberg.com/etfs/news.rss"
]
```


```
[{'Headline': 'Bubble Debate Drives Korean Retail Investors to Risky VIX Bets',
  'Link': 'https://www.bloomberg.com/news/articles/2025-10-19/bubble-debate-drives-korean-retail-investors-to-risky-vix-bets',
  'Article': 'Investors in South Korea looking to hedge their big US stock holdings or play their next wager are embracing a new type of trades: leveraged VIX bets.',
  'Date': '2025-10-19T00:00:00'},
 {'Headline': 'US Warns of ‘Imminent’ Attack by Hamas Against Palestinians',
  'Link': 'https://www.bloomberg.com/news/articles/2025-10-18/us-warns-of-imminent-attack-by-hamas-against-palestinians',
  'Article': 'The US State Department said it informed countries involved in the Gaza peace agreement that an attack by Hamas is being planned against Palestinians and that it would be a violation of the ceasefire deal.',
  'Date': '2025-10-18T22:09:03'},
 {'Headline': 'Protesters Oppose Trump in ‘No Kings’ Event in NYC',
  'Link': 'https://www.bloomberg.com/news/videos/2025-10-18/protesters-oppose-trump-in-no-kings-event-in-nyc',
  'Article': 'Demonstrators across the US turned out for what organizers said would be more than 2,600 “No Kings” protests across the US to express their opposition to President Donald Trump’s agenda. (Source: Bloomberg)',
```


LangGraph

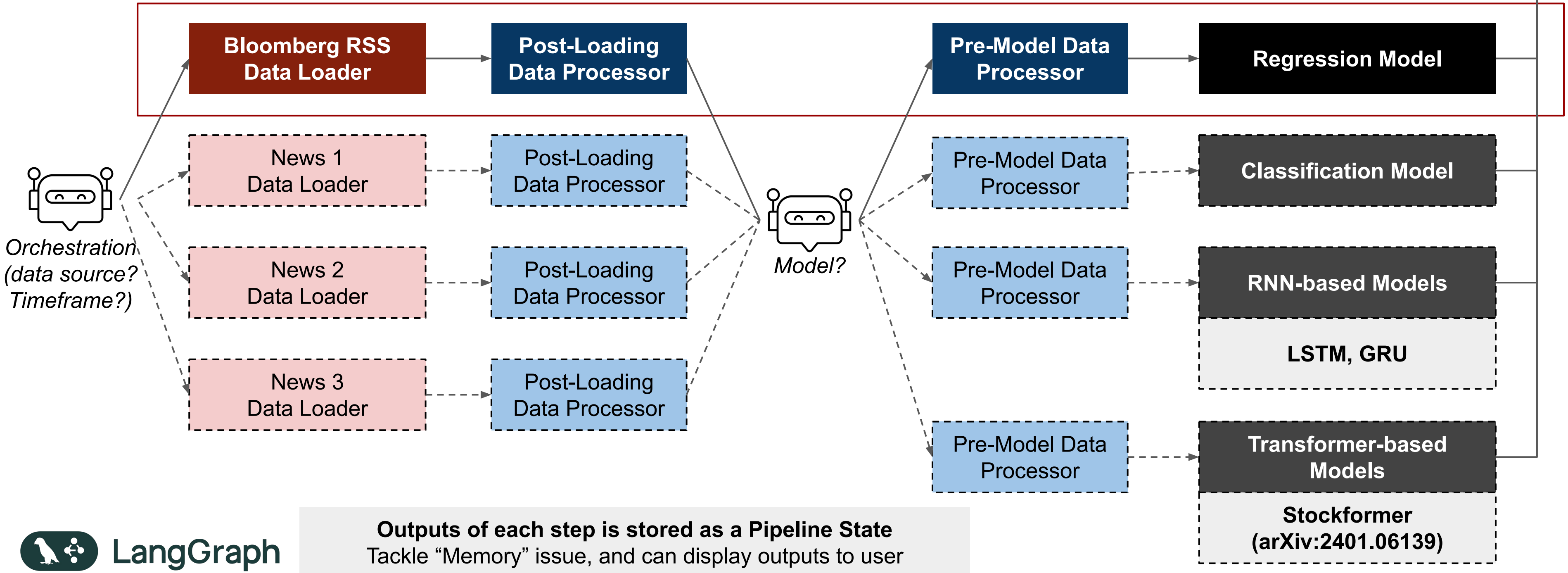


Current LangGraph Nodes and Edges

Inference using Agent - Target

- **New data sources, prediction models**
- **Multi-agent**
- **Using Agent for path planning (decision on next step)**

Streamlit



LangGraph

Outputs of each step is stored as a Pipeline State
Tackle “Memory” issue, and can display outputs to user

Demo

Conclusion

We introduce a framework to uses agents to do factor extraction from alternative data sources to drive agentic trading strategies.

Scalability

Cloud-ready:
- Training data is stored in highly efficient parquet files linked to Google Drive
- Can be stored and linked to s3 buckets

Real-time inference:
- Using data from live RSS feeds

Platform-Agnostic:
- Built Docker Image, can be deployed to Docker registry and deployed using Container Orchestration like Kubernetes

Conscious Design Choices

Task-specific BERT Variants: BERT
- Sentiment from FinBERT (specific for financial sentiment analysis)
- Industry classification from mDeBERTa-v3 (good at zero-shot NLI)

Combining Deterministic & Non-deterministic outputs
- Leverage BERT bidirectionality for classification and cost-saving
- LLM (decoder-based) for explanation generation OpenAI
- Classical ML models for factor modelling

Explainability of results

Data and prediction lineage LangGraph
- Ability to pinpoint for each industry/day pair, what are the actual news, sentiment scoring and explanation
- Especially important in financial industry