Text-Driven Market Movement Prediction

Final Project Report

Zitian

June 2025

Abstract

This study examines the predictive power of daily financial news headlines for next-day S&P 500 movements. We implemented a fully managed AWS pipeline—leveraging S3 for storage, Glue for ETL, Athena for SQL querying, and SageMaker for modeling—to extract both lexicon-based and FinBERT-derived sentiment features from headlines spanning 2008–2024. We benchmarked a range of models, including LSTM (regression and classification), logistic regression, tree-based ensembles (XGBoost, Gradient Boosting, Random Forest), and LightGBM under a 5-fold time-series split. All classifiers achieved AUCs between 0.50 and 0.52 and accuracies near 0.5, indicating minimal predictive signal in daily headlines alone. These results highlight both the challenges of news-driven market prediction and the effectiveness of an end-to-end cloud-native analytics workflow.

1 Introduction

Predicting equity market moves from unstructured text is challenging yet appealing. We use a Kaggle dataset pairing S&P 500 closing prices with SEC headlines (2008–2024) and a financial sentiment lexicon. Our goal: build an AWS-powered pipeline to extract daily sentiment features, train classifiers, and evaluate predictive power.

2 Data Sources

- **News & Price Dataset**
  Contains date, headlines, closing price (`cp`), and next-day return.
- **Financial Sentiment Lexicon**
  Lists financial terms with sentiment scores in $[-1, 1]$.

2.1 AWS Infrastructure

Below we describe each major stage in our AWS workflow, its purpose, and how it was implemented.

2.1.1 S3 Buckets and Data Ingestion

We rely on four dedicated S3 buckets:textdataproject-raw-snpnews-zitian,textdataproject-raw-lexicon-zitian, textdataproject-processed-features-zitian,textdataproject-query-results-zitian. To organize every stage of our pipeline. The first two ("raw") buckets hold the original Kaggle CSV and ZIP files for news headlines and the financial sentiment lexicon immediately after download via the AWS CLI (configured with our `TextDataProjectRole` in the Stockholm region). The "processed-features" bucket stores the Parquet outputs generated by Glue ETL jobs (daily sentiment metrics, tokenized headlines, etc.), and the "query-results" bucket is used

by Athena as its query staging location. This structure ensures clear separation between raw inputs, intermediate feature tables, and analytical query outputs, all within the same region for optimal performance and security.

2.1.2 IAM Roles and Permissions

To enforce least-privilege access across our data pipeline, we defined a single IAM role, `TextDataProjectRole`, which is assumed by AWS Glue, Athena, and QuickSight. Its trust policy explicitly permits the Glue, Athena, and QuickSight service principals to assume the role. In addition, we created individual IAM users with programmatic access keys mapped to this role for local development, ensuring that CLI operations and CI/CD pipelines inherit the same controlled permissions.

2.1.3 Glue Crawler and Data Catalog

We set up a Glue Crawler ("raw-news-crawler") to scan our raw S3 bucket (textdataproject-raw) on demand and automatically infer the schema of the uploaded CSVs, registering the resulting table directly into the Glue Data Catalog under the database `sentiment_db` with the prefix `raw_news_`. The crawler runs under the `TextDataProjectRole`, which grants it access to S3 and the Data Catalog. Once executed, it produced the table `sentiment_db.raw_news_table`, making the raw headlines and price data immediately queryable via Athena and downstream ETL jobs.

2.1.4 Glue ETL / CTAS to Parquet

To optimize query performance and support analytics at scale, we transformed the raw CSV tables into columnar Parquet files and materialized our daily feature table in S3. This was accomplished in two interchangeable ways: either by running a Glue ETL job— based on the default PySpark script template— which reads from `sentiment_db.raw_news_table` and writes out Parquet partitions to our processed-features bucket, or by issuing an Athena CTAS statement that selects from the raw table and writes directly in Parquet format to s3://textdataproject-processed-features-v2/. In both cases we confirmed that the output files appeared under the expected S3 path, ready for downstream querying and analysis.

2.1.5 Athena Query Configuration

To provide interactive, serverless SQL access over our Parquet feature tables and to feed downstream dashboards and notebooks, we configured Athena with a dedicated results bucket in the same region. After pointing Athena's query result location to `s3://textdataproject-query-results/` in `eu-west-1`, we verified connectivity by listing tables in `sentiment_db` and sampling rows from `daily_features_v2`. This setup ensures all query outputs and metadata are stored centrally, simplifying auditability and reuse across the project.

2.1.6 QuickSight Dashboard Integration

We set up an Amazon QuickSight dashboard to visualize daily sentiment metrics alongside price movements, sourcing data directly from our Athena view `v_daily_full_features` (with SPICE ingestion enabled for performance) under the `TextDataProjectRole`. The dashboard includes four concise visuals:

1. **Daily Price vs. Sentiment:** dual-axis line chart for closing price and lexicon/FinBERT sentiment.
2. **Quarterly Sentiment Trends:** grouped bar chart comparing lexicon total, average, dispersion, and FinBERT sentiment.

3. **Sentiment vs. Next-Day Return:** scatter plot of daily sentiment against next-day return with trendline.

4. **Sentiment Decile Returns:** bar charts showing average next-day return per sentiment decile for both lexicon and FinBERT scores.

2.2 Local Enhancement with Pandas

Once the daily Parquet files were available in S3, we used a SageMaker notebook to compute two complementary sentiment feature sets:

1. **Lexicon-Based Token Scoring:**

   - Loaded each day's headlines into a Pandas DataFrame.
   - Tokenized headlines using simple whitespace splitting and normalized to lowercase.
   - Merged tokens with our financial sentiment lexicon CSV (columns `word_or_phrase`, `sentiment_score`).
   - For each date, computed:
     - `daily_total_score` = sum of matched token scores;
     - `avg_sentiment` = mean of token scores;
     - `std_sentiment` = standard deviation of token scores (filled to 0 if only one token or one headline);

– `daily_pos_count` / `daily_neg_count` = counts of tokens with positive / negative scores.

2. **FinBERT Sentence-Level Analysis:**

- Initialized the Hugging Face pipeline:

```
from transformers import AutoTokenizer,
    AutoModelForSequenceClassification, pipeline
tokenizer = AutoTokenizer.from_pretrained("yiyanghkust/finbert-
    tone")
model     = AutoModelForSequenceClassification.from_pretrained(
    "yiyanghkust/finbert-tone")
finbert   = pipeline("sentiment-analysis", model=model,
    tokenizer=tokenizer, return_all_scores=True)
```

- Applied `finbert` to each raw headline to retrieve probabilities for `positive`, `neutral`, and `negative`.
- For each date, aggregated:
  – `finbert_pos_mean`, `finbert_neg_mean`: mean positive / negative probability;
  – `finbert_pos_std`, `finbert_neg_std`: standard deviation of those probabilities.

These sentiment features were then merged with daily price data (`daily_cp_from_lexicon`, `next_return`) and extended with simple time-series derivatives (`prev_return_pct`, `ma3_return_pct`) to construct the final modeling dataset.

3 Correlation Analysis of Features vs. Next-Day Return

To assess linear relationships between our engineered sentiment features and the target variable (`next_return`), we computed the pairwise Pearson correlation coefficients and visualized them in a heatmap (Figure 1).

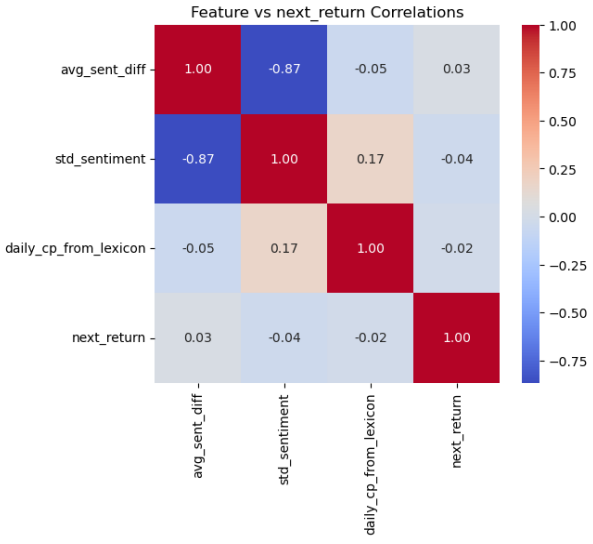

Figure 1: Pearson correlation matrix.

As shown, all off-diagonal correlations with `next_return` are very close to zero (–0.04 to +0.03), indicating almost no linear association between any single sentiment or price feature and the subsequent return. This low degree of linear dependence suggests that simple linear models may struggle to predict price movements from these features alone, motivating our exploration of non-linear and more complex modeling approaches in subsequent sections.

4 Modeling

Given the near-zero linear correlations observed, we focused on non-linear classifiers under a 5-fold `TimeSeriesSplit` scheme to respect temporal ordering. Table 1 summarizes the mean performance metrics for each model.

Table 1: Mean classification performance over 5 folds

Model	**AUC**	**Accuracy**	**F1 Score**
Logistic Regression (L2)	0.513	0.525	0.634
Gradient Boosting	0.502	0.471	0.376
Random Forest	0.502	0.494	0.490
XGBoost	0.508	0.505	0.499
LightGBM	0.522	0.512	0.565

Overall, despite using sophisticated non-linear algorithms, performance remained close to random guessing. This suggests that daily headline sentiment alone—especially given sparse daily counts and simplistic lexicon or sentence-level scores—does not contain sufficient signal to predict next-day S&P 500 movements.

5 Results & Discussion

Our end-to-end AWS pipeline successfully ingested raw headlines, engineered both lexicon-based and FinBERT-derived sentiment features, and benchmarked a suite of non-linear classifiers. Table 1 shows that the best performer—LightGBM—reached a mean AUC of 0.522, with accuracy 0.512 and F1 of 0.565. All other models hovered near chance level (AUC 0.502–0.513, accuracy 0.471–0.525), confirming that daily headline sentiment alone carries very weak predictive signal.

- **Data Sparsity and Noise:** Many trading days have only one or no headline, so lexicon and FinBERT scores are based on extremely limited text.
- **Coarse Sentiment Aggregation:** Simple token-sum and sentence-mean approaches may wash out important nuances, especially when headlines vary greatly in length and specificity.
- **Lack of Contextual Market Features:** Without incorporating price momentum, volatility measures, or macroeconomic controls, sentiment features cannot be interpreted in market context.
- **Measurement Error in Labels:** Next-day return is influenced by many factors (earnings releases, global events) beyond headline text; our binary up/down label may be too noisy for text-only models.

Although the LightGBM baseline slightly outperformed random guessing, its modest AUC (0.522) suggests that richer textual representations (e.g. transformer embeddings), multi-day context windows, and integration of technical/fundamental indicators are necessary to unlock any significant predictive power in financial news sentiment.