From Tweets to Trends: Predicting Stock Prices Using X Sentiment

Spring 2025

Boston University
Professor Seferlis
DS598 Big Data Engineering

Team Members:

Yuchen Li | nikkili@bu.edu
Syeda Aqeel | sharoo@bu.edu
Hsiang Yu Huang | huanghy@bu.edu

Contents

1 Introduction **4**
1.1 Business Applications of Sentiment-Driven Stock Forecasting 4

2 Data Architecture **4**
2.1 Medallion Architecture Overview 4
2.2 Data Sources 6
2.2.1 X/Twitter Data (via RapidAPI) 6
2.2.2 Why NVIDIA? 6
2.2.3 Stock Data (via Yahoo Finance) 6
2.2.4 Rationale for Predicting Trading Volume Over Price or Return 7

3 Data Pipelines **7**
3.1 Batch Processing 7
3.1.1 Tweet Pipeline (Bronze + Silver) 7
3.1.2 Stock Pipeline (Bronze + Silver) 10
3.1.3 EDA Summary 11
3.1.4 Gold Layer (Tweet & Stock Market Data Merge and Feature Enrichment via Spark Synapse) 14
3.1.5 Gold Aggregate: Dimensional Summaries for Visualization 16
3.2 Streaming Processing 16
3.2.1 Trigger Mechanism 16
3.2.2 Data Ingestion 17

4 Machine Learning Pipeline **17**
4.1 Model Training 17
4.2 Daily Prediction (6AM) 18

5 Feedback Loop: Backfill & Retraining **18**
5.1 Backfill Retrain (6:30 PM) 18
5.2 Conditional Retraining 19

6 Data Aggregation & Visualization **20**
6.1 Gold Aggregation 20
6.2 SQL Integration 20

7 Synapse Orchestration & Resource Management **21**
7.1 Scheduled Triggers 21
7.2 SQL Pool Management 21

8 Power BI Dashboard **23**
8.1 Overview & Key Metrics 23
8.2 Tweet & User Activity Visuals 23
8.3 Behavioral Ratios & Scores 24

8.4 Sentiment vs. Stock Volume Analysis 24
8.5 Viral Tweet Analysis 25
8.6 Monthly & Daily Filters 25

9 Challenges & Optimizations **25**

10 Conclusion **27**

11 References **27**

1 Introduction

Social media, particularly X (formerly Twitter), has become an influential platform for shaping market sentiment in real time. This project aims to predict stock price movements based on public sentiment expressed on X. By analyzing tweets related to stock market activity and modeling their relationship with stock price fluctuations, we aim to build a predictive framework using sentiment analysis and machine learning techniques.

1.1 Business Applications of Sentiment-Driven Stock Forecasting

- **Algorithmic Trading & Investment Strategy:** Institutional and retail traders can incorporate sentiment-driven forecasts into automated trading algorithms to exploit short-term market movements more effectively than traditional technical analysis.
- **Market Sentiment Monitoring for Risk Management:** Financial firms can identify upcoming volatility or investor panic early, allowing risk managers to hedge or adjust positions proactively.
- **Business Intelligence and Forecasting:** Helps investment managers, portfolio analysts, or financial consultants identify patterns, track influencers, and make data-driven forecasts.
- **Investor Relations & Corporate Communications:** Companies like NVIDIA can tailor their communication strategy based on sentiment trends to mitigate reputational risk and reinforce brand messaging.
- **Influencer and Marketing Strategy Evaluation:** Marketing teams and hedge funds can identify which influencers drive market perception, enabling better targeting or partnership strategies.
- **Quantitative Research & Financial Innovation:** Academic institutions, R&D labs, and financial startups can use it as a foundation to explore additional use cases like anomaly detection, price prediction, or sector-wide modeling.
- **Cost-Efficient Decision Systems:** This architecture supports real-time decision-making with minimal human intervention and cost-effective infrastructure.

2 Data Architecture

2.1 Medallion Architecture Overview

This project adopts the Medallion Architecture to organize data into structured layers—**Bronze**, **Silver**, and **Gold**—to support a modular, scalable, and real-time pipeline. Our goal is to use sentiment from X (formerly Twitter) to predict the **next-day stock volume** of NVIDIA Corporation (NVDA), leveraging both historical (batch) and real-time (streaming) data from two primary sources: **RapidAPI** and **Yahoo Finance**.

Figure 1: Project Architecture

Key Flow

- **Data Sources**
  - Tweets from RapidAPI's Twitter V2 API (batch & streaming)
  - Stock market data via Yahoo Finance (batch & streaming using `yfinance` in Synapse)
- **Ingestion Layer (Bronze)**
  - Batch tweets ingested via Azure Data Factory and stored in Blob Storage
  - Streaming tweets ingested via Function App (bypassing Event Hub for simplicity) and directly saved in Gold-ready format
  - Stock batch data fetched & transformed via Synapse notebooks; streaming stock pulled together with streaming tweets inside Function App
- **Transformation Layer (Silver)**
  - Tweets preprocessed via Data Factory into structured format and stored in Blob Storage
  - Stock data cleaned and aligned within Synapse notebooks
  - Silver datasets joined and aggregated in Synapse Spark, forming the Gold layer
- **Gold Layer**
  - Final dataset used for model training and prediction
  - Includes both batch-historical and streaming-enriched data
- **Visualization & Prediction**
  - Predictions made using Synapse MLlib
  - Aggregated results exposed via SQL External Tables, Fact and Dimension Tables
  - Visualized using Power BI with daily retraining triggered by error threshold

2.2 Data Sources

2.2.1 X/Twitter Data (via RapidAPI)

We collect tweet data using RapidAPI's Search Twitter V2 (Explore endpoint). The query is designed to capture posts with the hashtag `#$NVDA`, ensuring relevance to NVIDIA's stock. Each request is set to pull English-language content only (`lang:en`) and is refreshed daily by adjusting the `since` and `until` parameters. Example query looks like:

```
(#$NVDA) lang:en since:2024-04-13 until:2024-04-14
```

- The hashtag `#$NVDA` is a finance-specific tag commonly used to refer to NVIDIA's stock.
- Metadata includes tweet content, timestamp, likes, replies, reposts, etc.
- Due to cost and data limitations, our dataset spans from January 2024 to present.

2.2.2 Why NVIDIA?

We chose to focus on one stock–NVIDIA–for precision. Modeling a single company allows us to:

- Avoid noise caused by sector-wide variability
- Improve prediction accuracy
- Leverage unique industry patterns

We were particularly interested in capturing social sentiment impact around key events, such as the launch of DeepSeek–an AI platform speculated to affect NVIDIA's valuation. Initially seen as a threat due to its cost-effectiveness compared to NVIDIA's offerings, DeepSeek later drove demand for NVIDIA's GPUs, leading to stock recovery. This event highlighted how social media buzz directly influenced investor sentiment and trading volume, justifying our narrowed focus on NVIDIA and Twitter.

2.2.3 Stock Data (via Yahoo Finance)

Stock data is retrieved via the `yfinance` Python library, natively supported in Synapse notebooks. The following fields are collected:

- Open, High, Low, Close prices
- Volume (used as the prediction target)

Both batch (historical) and streaming stock data are retrieved and integrated with the corresponding tweet data.

2.2.4 Rationale for Predicting Trading Volume Over Price or Return

Predicting stock prices or returns is inherently challenging due to the non-stationary and highly volatile nature of price data, which often leads to unreliable forecasting outcomes. Based on expert guidance from Professor Peter Wysocki of Boston University's Questrom School of Business, we elected to shift our focus toward a more stable and interpretable target:

> *"The future daily trading volume of NVIDIA using changes in sentiment from Twitter."*

Trading volume offers a grounded and less volatile measure of market activity that still effectively captures investor attention and behavioral trends. This choice enhances the model's robustness and practical relevance in sentiment-driven forecasting.

3 Data Pipelines

3.1 Batch Processing

3.1.1 Tweet Pipeline (Bronze + Silver)

Bronze Layer (Ingestion via Azure Data Factory)
The Bronze Layer handles raw data ingestion from external APIs, preserving as much of the original structure and fidelity as possible. For this project, our focus is on collecting tweets related to NVIDIA using the RapidAPI Twitter V2 (Explore Endpoint). The challenge lies not just in querying the data, but in dealing with the API's pagination structure and inconsistent return behavior.

API Access & Query Construction
To extract tweets, we use a daily query structure like:

```
/search-v2?type=Top&count=500&query=(%23$NVDA)%20lang:en%20
   until:2024-04-14%20since:2024-04-13
```

Again, this ensures that we are hashtag-specific, english-only for consistency in downstream sentiment analysis, and data-scoped. Although we specify a `count=500`, in practice, each API call only returns ˜20 tweets. To collect more, we must paginate using the bottom cursor returned in the response.

Challenges with Cursor-Based Pagination

- Always returns a bottom cursor, regardless of whether additional tweets exist.
- Does not clearly indicate when the tweet set is exhausted.
- May limit historical data (especially tweets older than 7 days).

Because of this, custom logic is required to:

- Count the total tweets fetched.
- Detect when there are no more tweet entries (by checking if `result.timeline.instructions[0].ent` only contains top/bottom cursors).
- Track and store the bottom cursor for use in subsequent requests.

Azure Data Factory Pipeline Structure
The Bronze layer ingestion is implemented using ADF and structured as follows:

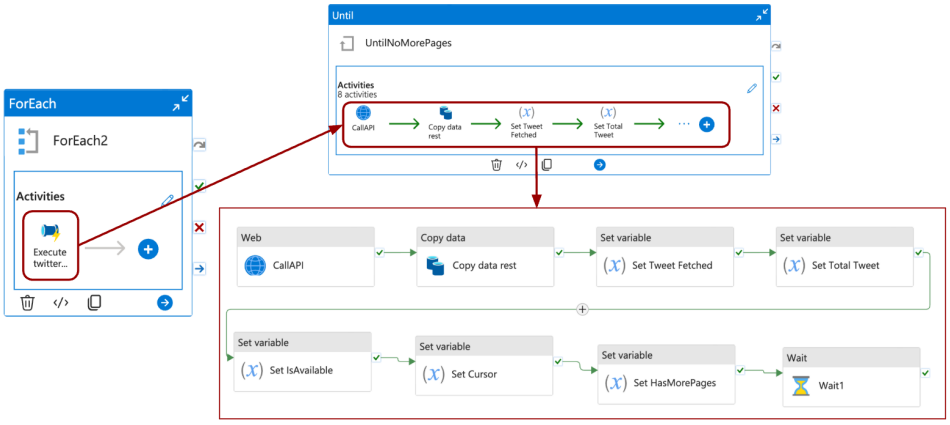

Figure 2: Tweet Data Bronze Layer Pipeline

- **Outer ForEach Loop**: Iterates over each target date based on a global parameter `dayAfter`, which defines how many days after the start date we wish to pull data for.
- **Inner Execute Pipeline**: Executes a pipeline named `UntilNoMorePages`, containing an `Until` activity that:
  - Repeats API calls until either:
    - The predefined tweet count is reached, or
    - No more tweets are available (as determined by inspecting the API response).
- **Inside the Until Loop**:
  - `CallAPI (Web Activity)`: Sends the HTTP GET request to RapidAPI.
  - `Copy Data`: Persists the returned tweet batch to Blob Storage (structured by date).
  - `Set Variables`:
    - `Tweet Fetched`: Number of tweets returned in the current call.
    - `TotalTweet`: Cumulative tweets collected for that date.

    - `Cursor`: Extracts and stores the bottom cursor.
    - `HasMorePages`: Checks if the cursor is valid and if additional entries exist.
    - `IsAvailable`: A flag that becomes false when only cursor entities are returned (meaning no actual tweets).
  - `Wait Activity`: Adds a buffer to avoid rate-limiting.

This robust looping structure ensures we efficiently and comprehensively capture tweet data under API constraints, storing it in a raw, date-partitioned format in the Bronze layer.

Silver Layer (Process via Azure Data Factory Data Flow)

The Silver Layer refines the semi-structured tweet data ingested during the Bronze stage by performing normalization, cleaning, and feature derivation. This transformation pipeline is implemented using Azure Data Factory's Mapping Data Flow and is executed on a monthly basis to optimize throughput and processing efficiency.

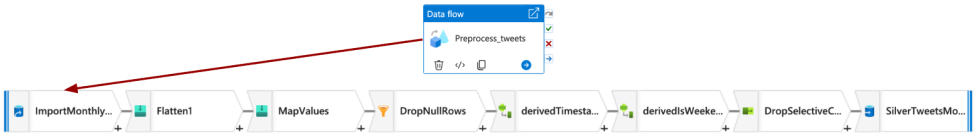

Figure 3: Tweet Data Silver Layer Pipeline

Each monthly data flow performs the following sequence of operations:

1. **ImportMonthlyTweets**
   This component ingests raw JSON files from Azure Blob Storage, specifically from the Bronze layer directory organized by month. These files contain nested Twitter metadata retrieved through RapidAPI's Search V2 endpoint.

2. **Flatten1**
   Due to the nested structure of the API response (e.g., `result.timeline.instructions`), this transformation flattens the array hierarchy, exposing the entries array for further extraction.

3. **MapValues**
   This step unrolls the entries array and maps relevant fields from both the user and tweet scopes:

   - **User-level attributes:** `user_id`, `is_blue_verified`, `followers_count`, `friends_count`, `account_created_at`, `listed_count`, `media_count`, `account_favourites_count`, `account_possibly_sensitive`
   - **Tweet-level attributes:** `rest_id`, `tweet_created_at`, `view_count`, `retweet_count`, `reply_count`, `quote_count`, `favorite_count`, `tweet_possibly_sensitive`, `full_text`

4. **DropNullRows**
   This step ensures downstream data integrity by removing rows containing null values in any of the critical columns listed above. It guarantees that only complete records are retained for enrichment and analysis.

5. **derivedTimestamp**
   This transformation creates new temporal fields from the existing data:

   - `tweet_created_at_date`: Extracted and cast to Date type.
   - `tweet_created_at_time`: Extracted and cast to String format.
   - `account_created_at`: Reformatted into a consistent Date type.

   These columns are essential for temporal aggregation and aligning with market activity windows.

6. **derivedIsWeekend**
   A binary flag (`is_weekend`) is generated based on the day of the week for each tweet, enabling identification of tweets posted during non-trading days.

7. **DropSelectiveColumns**
   This step removes intermediary or redundant columns (such as raw timestamps) that were only used for deriving structured values.

8. **SilverTweetsMonthly**
   The final cleaned and normalized dataset—comprising 20 carefully selected columns—is exported to a new Silver Layer container in Blob Storage. This version serves as the foundation for Gold Layer enrichment and subsequent modeling pipelines.

3.1.2 Stock Pipeline (Bronze + Silver)

Stock data is ingested and transformed using Synapse notebooks. Since Yahoo Finance provides structured historical data, both ingestion and processing are performed within Synapse before merging with tweets at the Gold layer.

Bronze Layer (Ingestion via Azure Synapse)
To obtain stock volume and other related information, we use the Python library yfinance to fetch historical data for NVIDIA stock (ticker: "NVDA"). By setting the start date to January 1, 2024, we retrieve daily stock data, including trading volume, opening price, closing price, high, and low. This data is then used within Azure Synapse for further analysis.

Parameter Setting
We set the timezone to New York time, as the largest trading market is located there, and it also aligns with Boston's timezone. Note that stock information will only be available on days when the market is open.

Silver Layer (Ingestion via Azure Synapse)
For the stock information silver layer transformation, we used Azure Synapse for data processing. In this step, we removed duplicate rows and transformed the Date column into a string time format to ensure alignment with the date format used in the tweet data. The processed data was then saved in Parquet format and stored in the silver layer.

3.1.3 EDA Summary

The Exploratory Data Analysis (EDA) phase was critical in understanding the structure, quality, and dynamics of the integrated sentiment and stock market datasets before proceeding with modeling. This section summarizes key visualizations and the insights they revealed regarding social sentiment, user behavior, and market activity.

1. **Market Reaction to DeepSeek Launch**

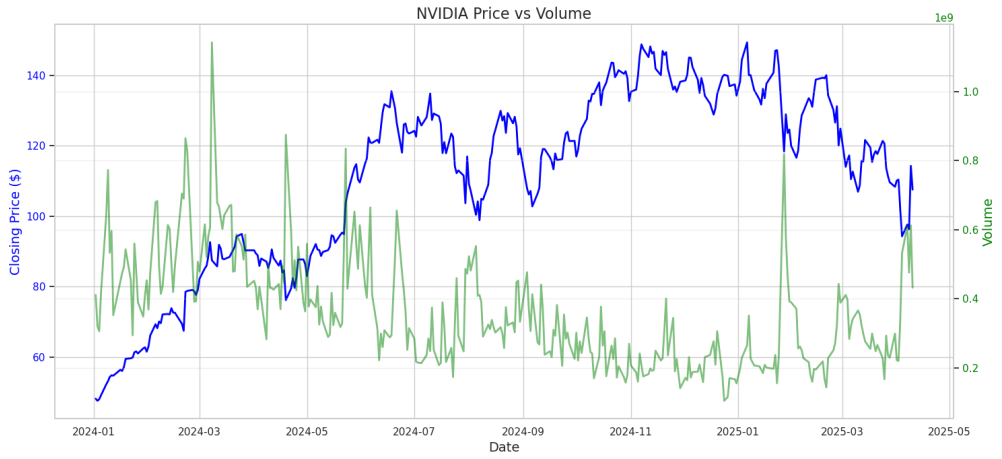

Figure 4: NVIDIA Price vs. Volume

In early 2025, NVIDIA experienced a notable decline in stock price accompanied by a sudden surge in trading volume. This market behavior closely aligned with the public response to the launch of *DeepSeek*, an AI platform initially perceived as a threat to NVIDIA's core business. The social media discourse surrounding the event reflected investor concerns, with a significant uptick in tweets carrying negative or speculative tones. Interestingly, as the market reassessed DeepSeek's reliance on NVIDIA's GPUs, the stock began to recover, highlighting how public sentiment can shift rapidly in response to evolving narratives. This episode underscores the importance of monitoring real-time sentiment as a leading indicator of market activity.

2. **Correlation Between Stock Metrics**

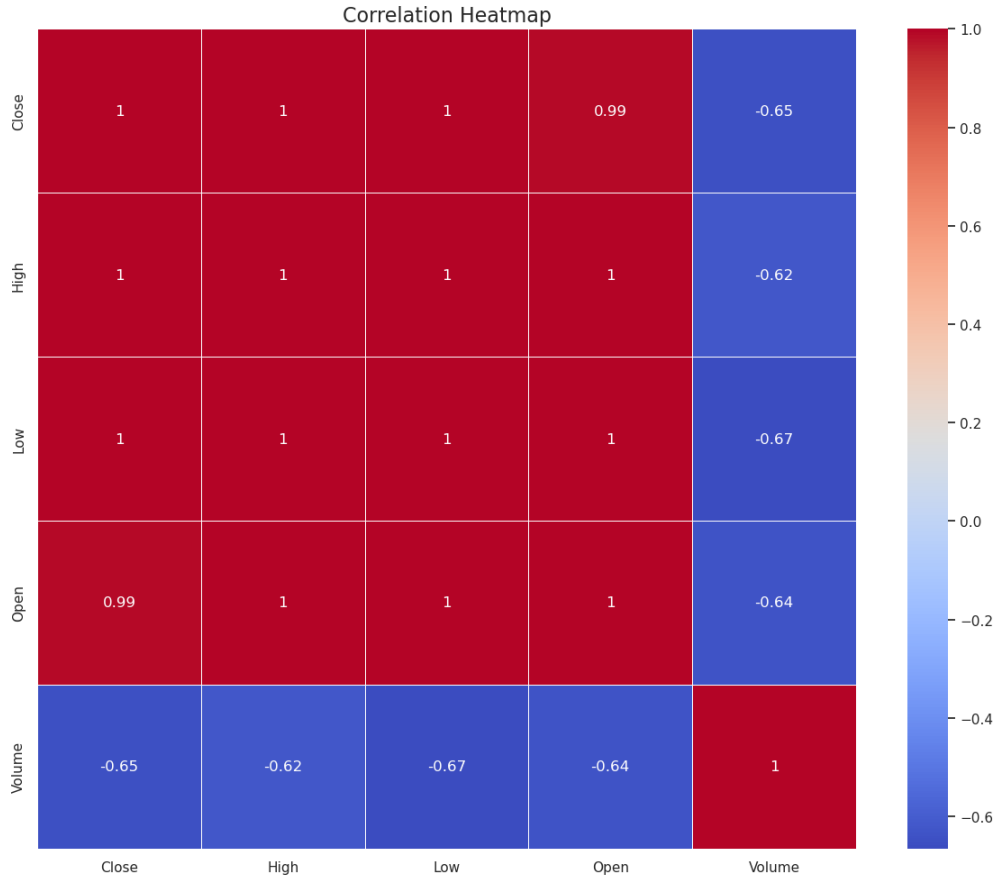

Figure 5: Correlation Heatmap

The correlation analysis explored relationships among key stock variables such as trading volume, closing price, sentiment scores, and tweet volume. The findings revealed a positive association between social media activity and stock trading volume, indicating that fluctuations in online sentiment often coincide with increased investor activity. Notably, while stock price remains challenging to predict due to its volatile and non-stationary nature, trading volume demonstrated stronger and more interpretable correlations with public engagement on X. This justifies the project's focus on predicting volume rather than price, enabling more stable and actionable forecasting outcomes.

3. **Influence of Verified Users on Engagement**

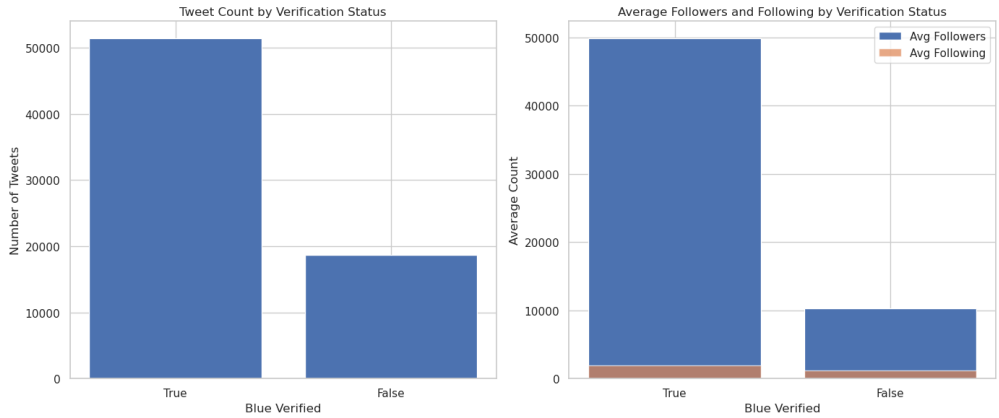

Figure 6: Tweet User Type Distribution

Tweets from blue-verified accounts—representing credible users or influencers—garnered significantly more engagement than those from unverified sources. This pattern reflects the trust and visibility associated with verified accounts, making their content more likely to be seen, liked, and shared. In the context of financial markets, these users often serve as information amplifiers, capable of swaying investor sentiment on a larger scale. Their disproportionate impact suggests that weighting tweet data by user credibility can enhance model performance and lead to more accurate forecasts.

4. **Favorites vs. Retweets and Replies**

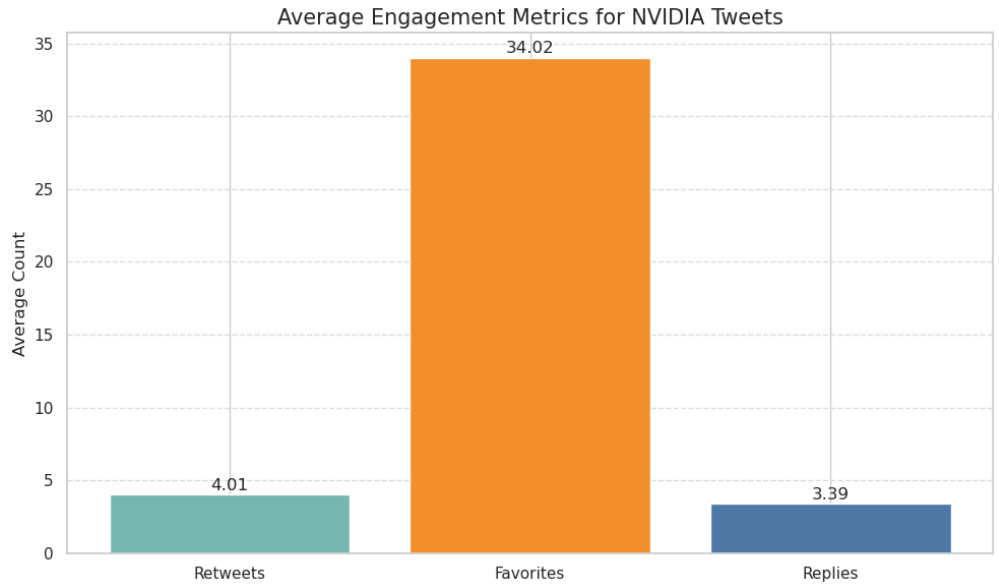

Figure 7: Average Engagement Metrics for NVIDIA Tweets

When examining user interaction with #NVIDIA tweets, *favorites* emerged as the dominant form of engagement, surpassing retweets and replies. This trend indicates

that users are more likely to passively endorse content than actively disseminate or comment on it. From a sentiment analysis perspective, the number of favorites can be interpreted as a proxy for approval or resonance with the tweet's message. In contrast, retweets and replies may carry more ambiguous sentiment and are often motivated by disagreement, humor, or unrelated discourse. This insight supports the use of favorites as a stronger signal when quantifying public mood in financial contexts.

5. **Most Commonly Co-Mentioned Stock Tickers**

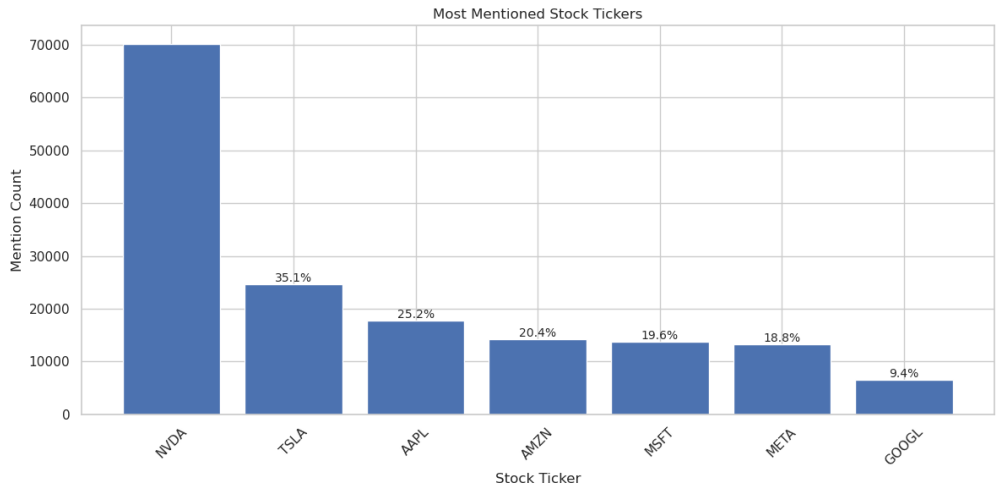

Figure 8: Most Mentioned Stock Tickers

The analysis of tweets containing the #NVIDIA hashtag revealed frequent co-mentions of other tech-related stock tickers such as `$TSLA`, `$AMD`, and `$MSFT`. This co-mentioning behavior reflects broader investor conversations about the technology sector, often linking companies based on industry developments, comparative performance, or thematic narratives like AI and chip manufacturing. Understanding these associations is valuable for identifying cross-influences between stocks and exploring multi-stock sentiment modeling. It also suggests that social media analysis for one company can provide spillover insights into its peers and competitors.

3.1.4 Gold Layer (Tweet & Stock Market Data Merge and Feature Enrichment via Spark Synapse)

The Gold Layer represents the culmination of the data transformation pipeline, integrating tweet-level sentiment and user engagement data with stock market statistics to form a machine-learning-ready dataset. This stage is implemented using Synapse Spark in the `TweetStockGold.py` notebook and stored in Azure Blob Storage.

Source Ingestion
The process begins by reading in silver-layer outputs from both data sources:

- **Tweet Data**: Loaded from monthly partitioned files in the `Tweet_Monthly_Silver` container.
- **Stock Data**: Daily records for NVIDIA stock—containing fields such as Open, Close, High, Low, and Volume—are also loaded from Blob Storage.

Sentiment Analysis
To quantify the sentiment of each tweet, the full text is first cleaned to remove URLs and extraneous whitespace. The cleaned content is then scored using the VADER Sentiment Analyzer, which returns a compound sentiment score ranging from -1 (very negative) to 1 (very positive). VADER is selected for its performance on social media data, offering nuanced scores rather than categorical labels, which is more informative for regression-based modeling and visualization.

Feature Engineering
In order to capture user behavior and tweet impact, a set of derived features is computed:

- **Engagement Metrics**:
  - `interaction_score` = weighted sum of engagement types (retweet, reply, quote, favorite) normalized by view count.
  - `favorite_ratio` = favorites / views
  - `reply_ratio` = replies / (retweets + 1)
- **User Attributes**:
  - `account_age_days`, `credibility_score` (based on blue verification, followers, and age), and `follower_activity_score` (likes + media + listed / age)
- **Categorical Flags**:
  - `is_viral` (if `interaction_score > 2000`), `is_new_account` (<1 year old), `is_influencer` (>130K followers), `is_holiday`, `day_of_week`

These features enrich the dataset with contextual signals critical for modeling stock activity.

Dataset Merge and Output
The tweet and stock datasets are merged on the `tweet_created_at_date` column. To support next-day prediction, a `next_available_volume` column is added by referencing the volume of the next trading day. Records from non-trading days (weekends and market holidays) are removed, and missing values (e.g., `view_count`) are imputed with default values. The final enriched dataset is written to the `Gold/dataset_updated/` directory in Blob Storage.

3.1.5 Gold Aggregate: Dimensional Summaries for Visualization

To facilitate business intelligence reporting and enable rich insights through Power BI, a set of aggregated summary tables is created in the `GoldAggregate.py` notebook. These aggregations offer both descriptive statistics and analytical views on the tweet and stock datasets.

Output Datasets

- **User Profile Table**
  Captures per-user metrics such as follower count, credibility score, and flags for influencer status and account age. This table enables profiling of active or influential users.
- **Daily Tweet Summary**
  Aggregates tweets by date, including metrics like average sentiment score, engagement totals, and number of viral tweets. This allows for trend and anomaly detection in sentiment and volume.
- **Market Summary per Day**
  Combines the tweet summary with daily stock data to support correlation analysis between social sentiment and market performance.
- **Viral Tweet Log**
  Contains only tweets flagged as viral. This is used for qualitative exploration of high-impact tweets and content strategy evaluation.
- **User Type Distribution**
  Provides daily breakdowns of tweet authors by account type—verified, influencer, or new account—offering insight into who is driving the conversation.
- **Top Users by Tweet Count**
  Highlights the most prolific contributors to the dataset, identifying high-activity users.
- **Top Users by Viral Tweets**
  Identifies users whose content goes viral most frequently, surfacing key influencers.
- **Influencer Monthly Engagement**
  Tracks engagement metrics for influencers month over month, allowing longitudinal analysis of influencer performance and impact.

Each dataset is exported to Blob Storage and made available for external table mapping within Synapse, supporting downstream analytics and Power BI dashboard integration.

3.2 Streaming Processing

3.2.1 Trigger Mechanism

A Function App is scheduled to run daily at 5:00 AM EST, collecting tweets and stock data.

3.2.2 Data Ingestion

Since Event Hub has a limit of 1 MB per message, handling hundreds of detailed tweets along with stock price data becomes challenging. To overcome this, we preprocess and aggregate the data within the Azure Function App, then store the results as a JSON file and directly ingested into the Blob Storage Gold layer, ensuring consistency with the predefined Gold schema. Before fetching any data, the function checks whether the market was open on the previous day and, if so, retrieves that day's data. Data is only collected on trading days. For instance, we use Friday's tweets and stock information to predict market trends for the following Monday. The time trigger function triggers every day at 5am (please refer to `StreamingStockVolume Function App`), ensuring the previous stock information has already been updated.

Tweet Process
For tweet ingestion, we apply the same logic as with batch data: fetching the top 20 tweets each time and updating the cursor until we reach the desired amount. Once collected, the data is processed in the Azure Function App, where tweet data is aggregated with corresponding stock information in preparation for prediction. After processing, the result is saved as a JSON file and uploaded to the `Medallion/Gold/Stream` folder in Blob Storage.

4 Machine Learning Pipeline

4.1 Model Training

To generate a prediction model capable of estimating NVIDIA's next-day trading volume based on tweet sentiment, we utilized the gold-layer batch data. The training pipeline was developed using Apache Spark MLlib within Azure Synapse.

The key steps are as follows:

- **Data Source:** The `dataset_updated` Parquet files from the gold layer were loaded from Azure Data Lake Storage.
- **Feature Engineering:** A combination of boolean casting (e.g., `is_viral`, `is_influencer`), one-hot encoding (e.g., `day_of_week`), and numerical scaling was applied. Features such as sentiment score, interaction ratios, and user credibility metrics were included.
- **Vector Assembly:** All features were consolidated using a `VectorAssembler` and standardized with `StandardScaler`.
- **Model Selection:** A `Random Forest Regressor` was trained with `Grid Search` to tune hyperparameters including the number of trees, max depth, and feature subset strategy.
- **Output:** The best-performing model pipeline was persisted to Azure Blob Storage for further use.

This model is retrained periodically based on prediction error thresholds calculated after each trading day (see Section 5).

4.2 Daily Prediction (6AM)

A scheduled Synapse trigger named `daily trigger` executes a PySpark notebook at 6:00 AM EST each day (please refer to `BlobStreamPredict.py`). The workflow is detailed below:

- **Input:** The notebook loads streaming data collected by the Azure Function App (executed at 5:00 AM) from Blob Storage.
- **Preprocessing:** The notebook applies the same transformation pipeline used in model training, ensuring consistency in feature encoding and scaling.
- **Prediction:** The stored best model is loaded, and predictions for the end-of-day stock volume are generated for each tweet record.
- **Output:** The predicted results are written to the `Prediction/Streaming` folder.

This daily inference pipeline enables real-time stock volume forecasting based on social sentiment trends gathered from X (formerly Twitter).

5 Feedback Loop: Backfill & Retraining

The predictive pipeline incorporates a daily feedback loop to evaluate model accuracy against actual stock performance and dynamically retrain the model if necessary. This enables the system to remain adaptive to market trends and changes in social sentiment behavior.

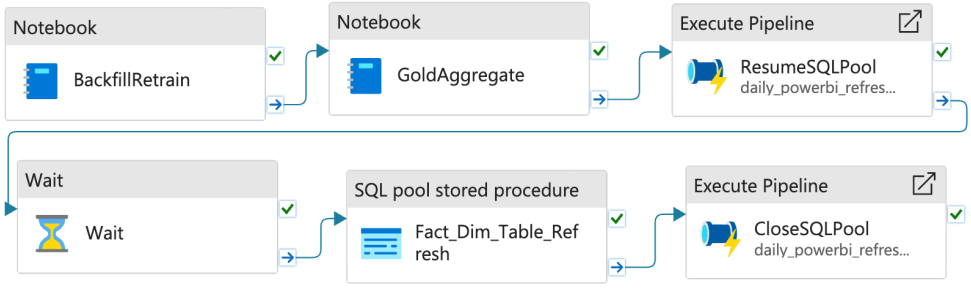

Figure 9: Daily Trigger Backfill & Retrain Pipeline

5.1 Backfill Retrain (6:30 PM)

At 6:30 PM EST, a Synapse pipeline triggers the `BackfillRetrain` notebook to backfill missing actual stock volume data and update both historical datasets and the machine learning pipeline.

Key steps include:

- **Actual Volume Retrieval:**
  The notebook uses the `yfinance` Python package to fetch the actual end-of-day stock volume for NVIDIA (ticker: `NVDA`). This ensures that model evaluation is based on verified market outcomes.

- **Stream Data Augmentation:**
  Streaming data for the day, previously stored at 5:00 AM, is reloaded. A new column `next_available_volume` is appended to this data, representing the real next-day value that the model was attempting to predict.

- **Historical Gold Update:**
  The enriched streaming data is appended to the existing gold-layer dataset (`dataset_updated`), extending the training set with fresh examples.

- **Prediction Archive:**
  The predicted result generated earlier at 6:00 AM is loaded from the corresponding path and appended to a cumulative prediction log for future error tracking and visualization.

This backfill operation ensures the model is consistently evaluated on the most recent real-world outcomes.

5.2 Conditional Retraining

Following the backfill, the notebook calculates the prediction error for the current day using the formula:

```
Error = Abs(actual - predicted) / actual
```

If the error exceeds a pre-defined threshold of 0.01 (i.e., a 1% relative error), the model is flagged for retraining.

Upon triggering, the notebook:

- Reloads the updated gold-layer dataset
- Re-executes the training pipeline using Spark MLlib
- Saves the newly trained best model and pipeline back to Blob Storage, replacing the previous version

This dynamic retraining mechanism enhances model responsiveness and ensures ongoing alignment with evolving market and sentiment behavior.

After retraining, the notebook also triggers the Gold Aggregate process (see Section 3.1.4) to regenerate the latest summary views, ensuring that Power BI visualizations remain accurate and up to date.

6 Data Aggregation & Visualization

To enable business insights and real-time monitoring of sentiment-driven stock behavior, the gold-layer data is further transformed into dimensional summary tables and visualized via Power BI dashboards. This process comprises two core components: Gold Aggregation and SQL Pool Integration.

6.1 Gold Aggregation

Following the Backfill & Retrain step, the `GoldAggregate` notebook is executed daily at 6:30 PM EST. It transforms the enriched gold-layer dataset into a series of pre-aggregated summary tables tailored for business intelligence reporting. These include user profiles, daily tweet summaries, market-level sentiment correlations, and influencer engagement logs.

Each aggregated output is saved as a Parquet file and registered as an external table in the Synapse dedicated SQL pool for downstream consumption.

(For a breakdown of the individual tables, see Section 3.1.4.)

6.2 SQL Integration

To maintain synchronization between the gold-layer aggregates and downstream dashboards, all aggregated Parquet datasets are linked to External Tables in Synapse SQL via the `CREATE EXTERNAL TABLE` syntax. On top of these external tables, the system defines a set of Fact and Dimension Tables, populated using a stored procedure named `sp_DailyRefresh`.

Stored Procedure Workflow:

- Truncate and refresh each Fact Table:
  - `FactDailyTweet`, `FactViralTweetLog`, `FactMarketSummaryPerDay`, etc.
- Insert from External Tables into corresponding Fact Tables.
- Columns are explicitly cast as needed (e.g., truncating `full_text` to `NVARCHAR(500)` or converting `view_count` to `INT`).
- `FactInfluencerMonthlyEngagement` includes additional column transformation logic, such as concatenating year and month into a date-formatted `year_month` column.

This stored procedure is executed daily after aggregation as part of the pipeline sequence. It ensures that Power BI dashboards always reflect the latest enriched and summarized data without manual intervention.

7 Synapse Orchestration & Resource Management

7.1 Scheduled Triggers

We set up an automated prediction pipeline for our streaming data, consisting of the four stages described earlier. The schedule is illustrated in Figure10: Function App (5AM), Daily Prediction (6AM), Backfill (6:30AM), and Power BI Sync (7:50AM–8:10AM).

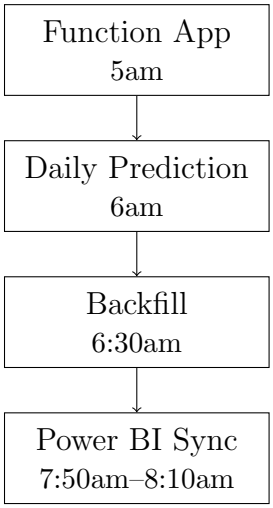

Figure 10: Daily processing pipeline for stock prediction

7.2 SQL Pool Management

To optimize cost efficiency while supporting scheduled dashboard updates, we implemented automated SQL pool management using Azure Synapse Pipelines. Since dedicated SQL pools incur compute costs when active, the pipeline intelligently controls the SQL pool lifecycle—resuming, checking state, and pausing—based on specific execution needs.

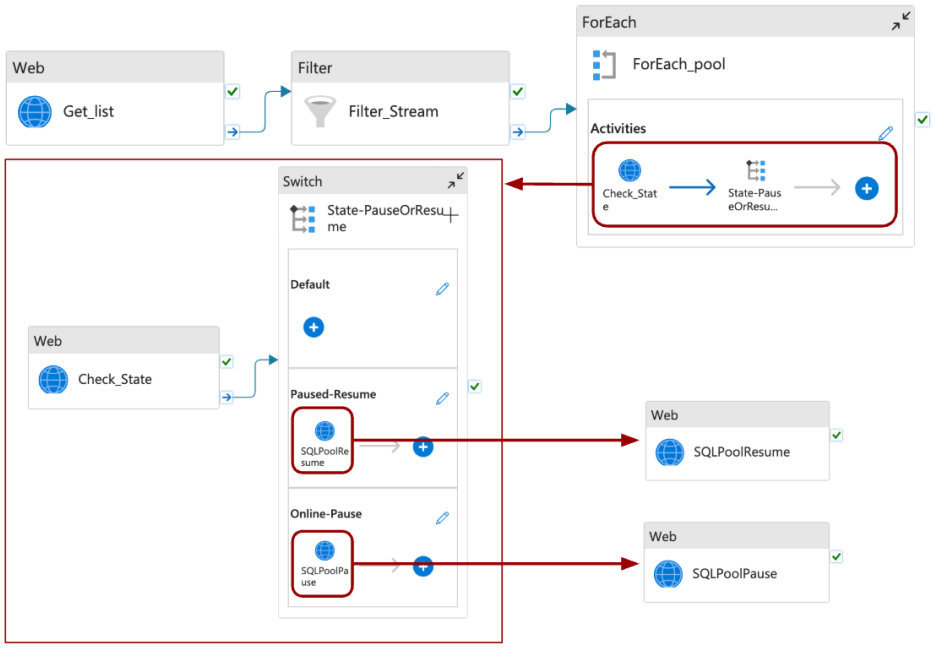

Figure 11: Scheduled Dedicated SQL Pool Resume/Pause Pipeline

Pipeline Logic Overview:

- **Resume SQL Pool**
  Before any downstream task (e.g., executing the `sp_DailyRefresh` stored procedure or enabling Power BI refresh), a Web Activity is used to issue a `POST` request to the Azure Management API to resume the SQL pool if it's paused. This is constructed using parameterized expressions within the pipeline.

- **Check Pool Status**
  A follow-up Web Activity queries the SQL pool's state (`Paused`, `Resuming`, `Online`, etc.) using a `GET` request. The response is parsed and evaluated through a Switch activity to handle different pool statuses. If the pool is not ready (`Paused`, `Resuming`), the pipeline can optionally wait and recheck.

- **Pause SQL Pool**
  After all data transformations and Power BI refreshes are completed, a final Web Activity issues a `POST` request to pause the SQL pool. This ensures that compute resources are only active during necessary windows (e.g., between 7:50 PM and 8:10 PM EST for Power BI daily refresh).

This modular and automated SQL pool orchestration ensures minimal operational cost without sacrificing data freshness or dashboard reliability.

8 Power BI Dashboard

To translate the processed data and machine learning outcomes into actionable insights, a dynamic Power BI dashboard was developed. This dashboard integrates with the SQL Pool via external and fact/dimension tables, refreshed daily using an orchestrated pipeline.

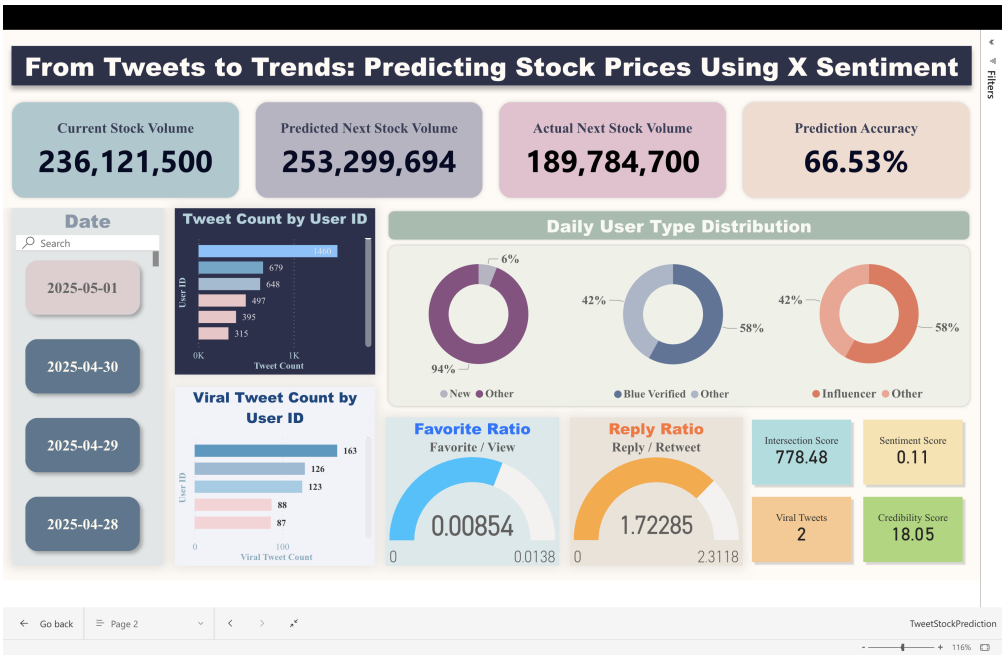

Figure 12: Power BI Dashboard Part 1

8.1 Overview & Key Metrics

The top-level KPIs summarize model performance and stock behavior:

- **Current Stock Volume:** Real-time volume from Yahoo Finance (same day).
- **Predicted Next Stock Volume:** Model-predicted volume for the following day.
- **Actual Next Stock Volume:** Real observed value retrieved the next evening.
- **Prediction Accuracy:** Computed as $1 - \left|\frac{\text{predicted}-\text{actual}}{\text{actual}}\right|$.

These metrics enable quick assessment of model reliability over time.

8.2 Tweet & User Activity Visuals

The dashboard offers granular insights into user engagement on X (formerly Twitter):

- **Tweet Count by User ID** and **Viral Tweet Count by User ID** identify highly active or influential contributors.

- **Daily User Type Distribution** uses pie charts to break down users into:
  - New vs. Established
  - Verified vs. Unverified
  - Influencer vs. General Public

  This classification helps assess credibility and influence dispersion in the dataset.

8.3 Behavioral Ratios & Scores

Several derived features are visualized to reflect engagement patterns:

- **Favorite Ratio:** Likes divided by views.
- **Reply Ratio:** Replies divided by retweets (activity depth).
- **Interaction Score**, **Sentiment Score**, **Credibility Score**, and **Viral Tweet Count** are also surfaced for deeper interpretation of sentiment-driven market dynamics.

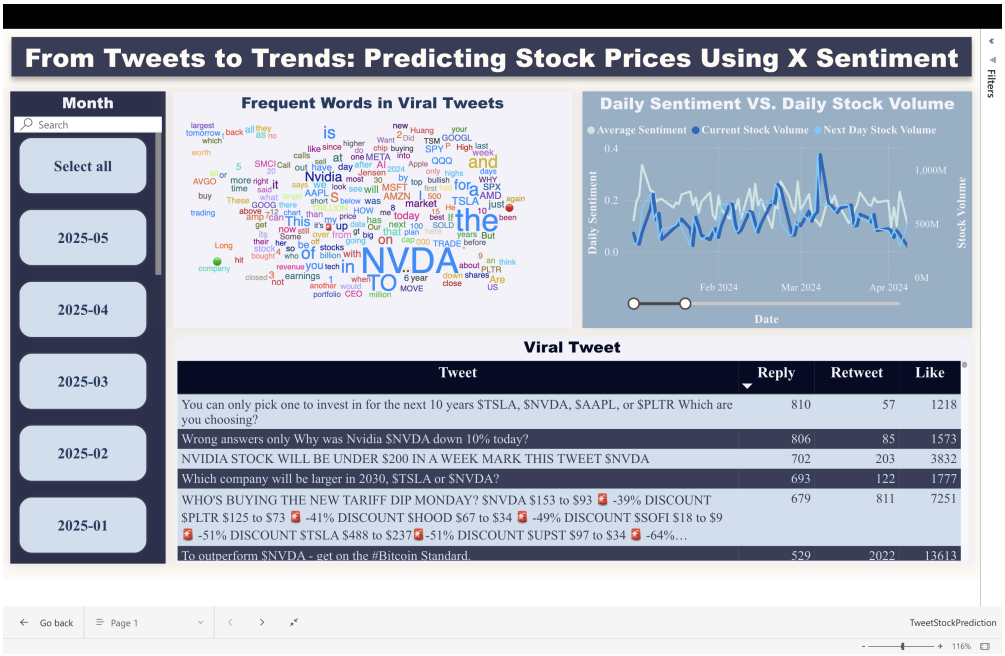

Figure 13: Power BI Dashboard Part 2

8.4 Sentiment vs. Stock Volume Analysis

A time-series line chart juxtaposes:

- Daily average sentiment score (from tweet content).
- Current stock volume.

- Next-day stock volume.

This comparison highlights sentiment–market behavior relationships across time.

8.5 Viral Tweet Analysis

The dashboard includes:

- A **Word Cloud** of most frequent terms in viral tweets—highlighting trending topics and brand perception.
- A **Viral Tweet Log Table** with engagement stats (replies, retweets, likes), allowing manual inspection of sentiment impact.

8.6 Monthly & Daily Filters

Interactive slicers for **Date** and **Month** support drill-down analysis, enabling users to track temporal shifts in tweet behavior and trading volume.

9 Challenges & Optimizations

- Data skew in modeling
  - One of the primary challenges in building our machine learning model was handling data skew—especially in engagement metrics like retweets and favorites. The distribution was heavily right-skewed, with most tweets receiving minimal engagement while a few went viral. This imbalance influenced model training by overemphasizing high-interaction outliers. We mitigated this by applying log transformations and scaling strategies, and by tuning the model to balance between common and rare engagement patterns.
- SQL pool cost management

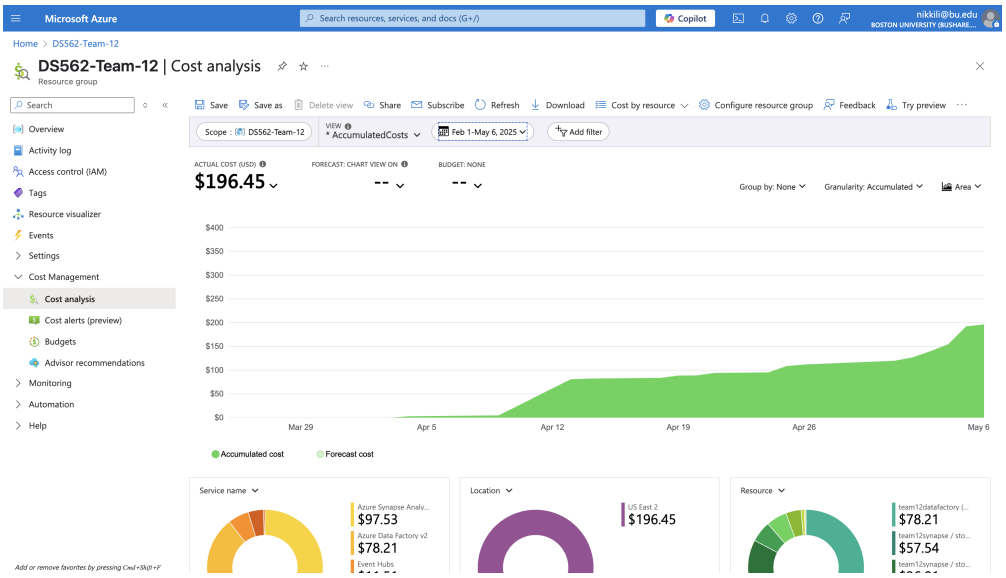

Figure 14: Cost Management

  - Azure Synapse's dedicated SQL pools incur significant compute costs when active, even during idle periods. Without proper control, our daily refresh pipelines and Power BI integrations could lead to unnecessary expenditure. We addressed this by building an automated SQL pool orchestration mechanism: the pipeline resumes the pool before heavy tasks, checks its availability state, and pauses it after processing. This ensured resources were only active when needed, substantially reducing operational costs.

- Efficient API pagination
  - The RapidAPI Twitter V2 endpoint required careful handling due to its opaque pagination system. Unlike typical APIs, it always returns a bottom cursor—even when no more tweets exist—and does not signal when the dataset is exhausted. We had to implement custom logic in Azure Data Factory using an Until loop that tracked the number of tweets fetched, evaluated cursor validity, and terminated gracefully when only navigation cursors remained. This prevented infinite loops and minimized redundant API calls.

- Feedback loop tuning
  - Our feedback loop, which compares predicted and actual stock volumes, required tuning to trigger retraining only when meaningful deviations occurred. Setting the error threshold too low would lead to unnecessary retraining and overfitting, while setting it too high could let model drift go unchecked. After experimentation, we set a 1% relative error threshold. This balance allowed the model to remain adaptive to new patterns without retraining excessively.

10 Conclusion

This project successfully integrates social media sentiment and stock market behavior to support predictive modeling, demonstrating the feasibility of using real-time social sentiment for financial forecasting. Leveraging the Azure cloud platform, we developed a scalable end-to-end pipeline that collects, processes, analyzes, and visualizes data with near real-time responsiveness. The system's modular architecture and dynamic feedback loop ensure continuous learning and adaptability to market changes.

For future work, we see several opportunities to enhance our system. First, incorporating weekend tweets could improve the prediction of Monday's stock volume by capturing more timely and relevant sentiment signals. Additionally, if we could access the total number of NVDA-related tweets on X, this metric could serve as a valuable input feature to better model tweet engagement and market impact. We also plan to explore anomaly detection to identify unusual behavior in either tweet or stock patterns, and to extend our analysis to support multiple stocks, allowing for broader market insights. Finally, integrating more advanced natural language processing models could lead to more accurate sentiment interpretation by better understanding the context and tone of tweets.

11 References

References

[1] Patel, Khushbu, and Anjali B. Bhesaniya. "Analysis and Prediction of Stock Market Using Twitter Sentiment and DNN." *ResearchGate*, 2019. https://www.researchgate.net/publication/336670851_Analysis_and_Prediction_of_Stock_Market_Using_Twitter_Sentiment_and_DNN

[2] Tripathy, Animesh, et al. "Stock Market Prediction Using Twitter Sentiment Analysis." *ResearchGate*, 2021. https://www.researchgate.net/publication/354308107_Stock_Market_Prediction_Using_Twitter_Sentiment_Analysis

[3] Ranco, Gábor, et al. "The Effects of Twitter Sentiment on Stock Price Returns." *ResearchGate*, 2015. https://www.researchgate.net/publication/282049046_The_Effects_of_Twitter_Sentiment_on_Stock_Price_Returns

[4] Salas-Zárate, María del Pilar, et al. "Stock Market Forecasting Using Sentiment Analysis: A Review." *Electronics*, vol. 11, no. 20, 2022, p. 3414. MDPI. https://www.mdpi.com/2079-9292/11/20/3414

[5] Prabhu, Ramesh. "Machine Learning for Stock Trading: Unsupervised Learning Techniques." *Analytics Vidhya*, Medium, 2019. https://medium.com/analytics-vidhya/machine-learning-for-stock-trading-unsupervised-learning-techniques-2be85e553361

[6] Bollen, Johan, et al. "Twitter Mood Predicts the Stock Market." *Procedia Computer Science*, vol. 1, no. 1, 2010, pp. 2231–2240. *ScienceDirect*. https://www.sciencedirect.com/science/article/pii/S1877050918308433

[7] Atef, Kerolos. "Stock Market Prediction Using Sentiment Analysis of Twitter." GitHub, https://github.com/KerolosAtef/Stock-market-prediction-using-sentiment-analysis-of-twitter/tree/main

[8] Microsoft. "Pause and Resume Dedicated SQL Pools Using Pipelines in Azure Synapse Analytics." *Microsoft Learn*. https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/how-to-pause-resume-pipelines