# A Data-Driven Approach to Stock Price Prediction

Shashidhar Babu Pasupuleti Venkata Durga
*M.S in Data Analytics*
*San Jose State University*
San Jose, USA

Vineeth Rayadurgam
*M.S in Data Analytics*
*San Jose State University*
San Jose, USA

***Abstract*—This project presents a Stock Price Prediction System that leverages historical financial data from the Yahoo Finance (yfinance) API to forecast future stock prices. The system is implemented using Apache Airflow for orchestrating automated data pipelines and Snowflake as the cloud-based data warehouse for storage and processing. The pipeline fetches daily stock market data for selected companies and ingests it into Snowflake, where machine learning models are deployed to predict stock prices for the next seven days. Snowflake ML functions are utilized for forecasting, ensuring seamless integration within the database. The automation of data ingestion, transformation, and forecasting allows for efficient, real-time stock price analysis, providing valuable insights for financial decision-making. This project demonstrates a scalable and robust approach to financial data analytics, integrating big data engineering with machine learning for enhanced stock market predictions.**

.

## I. PROBLEM STATEMENT

Accurately predicting stock prices is a critical challenge in financial markets, impacting investment decisions, risk management, and trading strategies. This project aims to develop a Stock Price Prediction System that forecasts stock prices using historical data and technical indicators.

The system integrates yfinance API for real-time and historical stock data retrieval, Snowflake for structured storage and processing, and Airflow for automated ETL and workflow orchestration. A database and data pipelines are essential to:

- Efficiently store and retrieve large volumes of stock data for analysis.
- Automate data extraction, transformation, and loading (ETL) processes.
- Ensure seamless updates of stock prices and forecasts at scheduled intervals.
- Enable machine learning-driven predictive modeling for informed decision-making.

By leveraging automation and machine learning, this system aims to provide accurate, scalable, and real-time stock price predictions for investors and analysts.

## II. SOLUTION REQUIREMENTS

### A. Historical Data

The system will use the yfinance API to obtain historical data for the last 180 days and real-time stock price data, including open, close, high, low prices, and trading volume from at least two companies (e.g., GOOGL, AAPL). This data will be stored in Snowflake, which will act as the system's database and data warehousing solution. By leveraging Snowflake's scalable architecture, the system can efficiently process large datasets and enable fast querying for financial analysis.

### B. Modeling Requirements

For forecasting, the system will utilize Snowflake's built-in ML capabilities to train time-series models on the stored data. This provides both short-term and long-term stock price predictions. In addition, the system will also implement feature engineering to improve prediction accuracy. Data preprocessing steps, including handling missing values, normalization, and feature scaling, will be incorporated to ensure high-quality model inputs.

### C. System Capabilities

The system will predict stock prices for user-defined time frames (e.g., next 7+ days) and provide confidence scores to indicate prediction reliability. Users will be able to visualize historical stock prices, trend analyzes, and technical indicators to aid in investment decision making. The entire ETL process, including fetching stock data from yfinance, storing it in Snowflake, running ML models, and updating predictions, will be automated using Airflow DAGs. This ensures seamless scheduling, real-time updates, and efficient model retraining.

### D. Limitations

The system's accuracy may decrease during periods of high market volatility or unexpected global events, as it primarily relies on historical data for predictions. Additionally, the performance of the system will depend heavily on the quality of the data retrieved from the yfinance API. Data quality issues such as missing values, gaps in yfinance data, or API rate limits could impact performance. The models built within Snowflake's AI/ML framework may also have limitations regarding model flexibility and customization.

### E. Use Case

The system is designed for financial analysts, investors, and portfolio managers who require automated stock price predictions, trend analysis, and technical insights. Users can enter stock tickers like 'AAPL' or 'GOOGL' to retrieve historical and real-time data, and access stock forecasts. The automated Airflow pipeline will ensure up-to-date stock data,

model retraining, and result visualization. Casual traders can also benefit from simplified reports and visualizations, helping them make data-driven investment decisions.

## III. Functional Analysis

The Stock Price Prediction system consists of several interconnected components, each playing a crucial role in achieving the goal of forecasting future stock prices. These components include data acquisition, storage, processing, machine learning, automation, and workflow orchestration. A key feature of this system is the usage of Directed Acyclic Graphs (DAGs) in Airflow, which manage task execution and ensure system efficiency. Below is an in-depth analysis of how these components interact with the database, data pipeline, and DAGs.

### A. Data Acquisition

The first functional component of the system is data acquisition, which involves fetching historical and real-time stock data from yfinance API. Users can specify stock tickers and time periods to trigger data retrieval. Once the data is retrieved, The ETL pipeline is orchestrated using Airflow DAGs, ensuring data extraction, transformation, and storage in Snowflake occurs. Using DAGs, the system ensures that the acquisition of new stock data happens in a predefined repeatable sequence at scheduled intervals or in response to specific user requests. Ensuring Idempotency using SQL transactions to prevent duplicate entries.

### B. Database and Storage Requirements

Snowflake serves as the primary storage and data warehousing solution, where both raw and processed data are stored. The data retrieved from yfinance API is structured in Snowflake to organize stock prices, technical indicators, and other key features required for forecasting.

Schema for the table:

- stock_symbol:Stock ticker symbol (e.g.,'AAPL', 'GOOGL')
- date: Date of stock price entry
- open: Opening price
- close: Closing price
- low: Lowest price of the day
- high: Highest price of the day
- volume: Number of shares traded

DAGs play a critical role in ensuring that data is regularly updated and stored in Snowflake. The data pipeline is automated through Airflow DAGs, which trigger periodic updates by fetching new data from yfinance at scheduled intervals.

### C. Forecasting Pipeline and Data pre-processing

After the stock data is loaded into Snowflake, the system performs data processing and feature engineering which are essential for enhancing the accuracy of the forecasting models. Data pre-processing steps like normalization, handling missing values, and scaling are also performed to ensure the data is suitable for machine learning. These data processing tasks are automated using DAGs in Airflow. Each DAG defines a sequence of tasks, such as fetching the data from Snowflake, applying transformations, creating features, and loading the processed data back into Snowflake. This automated orchestration ensures that data processing occurs seamlessly whenever new data is fetched, keeping the system updated with the latest stock information.

### D. Automation and Orchestration

The system is fully automated using Airflow DAGs, eliminating the need for manual intervention. Airflow manages task automation through the use of DAGs. Each component of the system, from data acquisition to machine learning, is represented as a task in a DAG. These DAGs control the flow of the entire pipeline, ensuring tasks are executed in the correct sequence and at scheduled intervals. Each DAG represents a distinct process. where one DAGs process is daily stock data retrieval from yfinance and other DAGs process is ML model training and forecasting updates.

## IV. Figures and Tables

This section outlines the table structures used in the Stock Price Forecasting system, provides examples of Python code utilized in data processing and machine learning, includes relevant SQL queries for data retrieval and manipulation in Snowflake, DAG implementation in Airflow and presents screenshots of the system interface.

### A. Overall Structure

The following block diagram explains the flow of data throughout the process. Initially the data would be extracted from yfinance API to Snowflake using ETL pipeline build by Airflow. Then the data would be processed and data would be used for machine learning training purposes which helps in forecasting new stock price values. The predictions will be stored back into Snowflake.

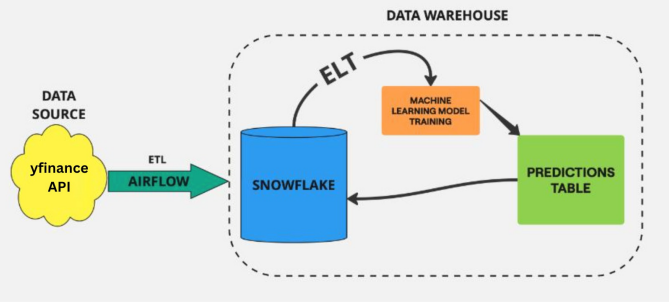


Fig. 1. Flow of Data

### B. Table Structure

Stock prices table is created with the raw data obtained from the Yfinance API. The ETL table contains the following columns 'date [Date], open [Float], high [Float], low [Float], close [Float], volume [Number] and symbol [Varchar]' of

the company. Forecast table is created to store the future stock prices generated by machine learning model. It has 'series [Variant], TS [Timestamp], forecast [Float], lower bound [Float] and upper bound [Float]' of the company.

```
cursor.execute("USE DATABASE stock_price")
cursor.execute("USE SCHEMA raw_data")

# Create or replace the table
create_table_query = """
CREATE OR REPLACE TABLE stock_prices (
    date DATE,
    open FLOAT,
    high FLOAT,
    low FLOAT,
    close FLOAT,
    volume BIGINT,
    symbol VARCHAR,
    PRIMARY KEY (date, symbol)
);
"""
cursor.execute(create_table_query)
print("Table created successfully.")
```

Fig. 2. SQL Query to create Stock Price Table in ETL DAG.

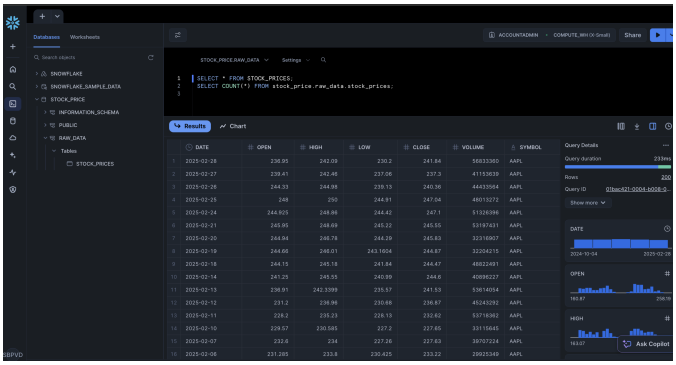

Fig. 3. Stock Price Table after ETL DAG triggered.

```
CREATE_VIEW_SQL = """
CREATE OR REPLACE VIEW STOCK_PRICES_v1 AS SELECT
    to_timestamp_ntz(DATE) as DATE_v1,
    CLOSE,
    SYMBOL
FROM STOCK_PRICES;
"""

CREATE_FORECAST_MODEL_SQL = """
CREATE OR REPLACE SNOWFLAKE.ML.FORECAST lab_1_forecast(
    INPUT_DATA => SYSTEM$REFERENCE('VIEW', 'STOCK_PRICES_v1'),
    SERIES_COLNAME => 'SYMBOL',
    TIMESTAMP_COLNAME => 'DATE_v1',
    TARGET_COLNAME => 'CLOSE',
    CONFIG_OBJECT => { 'ON_ERROR': 'SKIP' }
);
"""
```

Fig. 4. SQL Query to create Forecasting table in Forecasting DAG.

### C. DAGs

Directed Acyclic Graphs (DAGs) play a crucial role in orchestrating workflows within the Stock Price Prediction system. We have implemented 2 DAGs for the stock forecasting system, 'ETL' DAG and 'Forecasting' DAG.

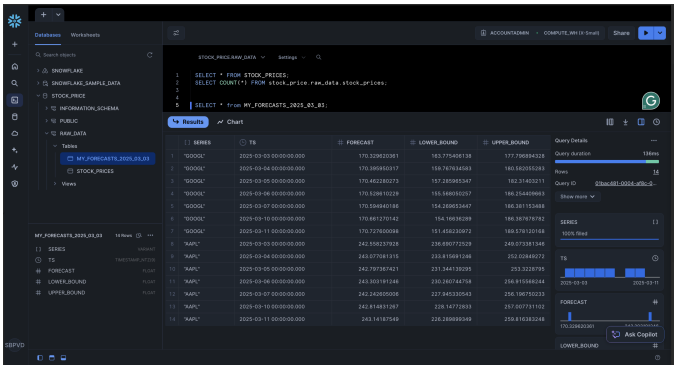

Fig. 5. Table showing Predicted prices of stocks 'AAPL' and 'GOOGL'.

### D. *ETL DAG*

The ETL DAG in Airflow, named 'Lab_1_DAG', is designed to automate the daily stock data pipeline for 'AAPL' and 'GOOGL' stocks using Snowflake and yfinance API. The DAG consists of tasks such as creating a Snowflake table, fetching stock data, and loading the data into the table. Each task is executed sequentially, with dependencies clearly defined in the DAGs graph view. The DAG can be manually triggered and monitored through the Airflow UI, which provides detailed logs and status updates for each task. This setup ensures efficient and reliable data processing for stock analysis.

### E. *Forecasting DAG*

The Forecasting DAG (stock_price_forecasting) in Airflow is designed to automate stock price prediction using Snowflake's machine learning capabilities. The DAG begins by setting up the Snowflake environment and inspecting the training data. It then creates a view (STOCK_PRICES_v1) to pre-process the data and trains the forecasting model using 'SNOWFLAKE.ML.FORECAST', focusing on the 'CLOSE' column as the target. The DAG generates predictions for the next 7 days, stores them in a table (My_forecasts_2025_03_03), and merges these predictions with historical data. Finally, it evaluates model accuracy using SHOW_EVALUATION_METRICS() and analyzes feature importance with EXPLAIN_FEATURE_IMPORTANCE(). This pipeline ensures a seamless workflow for stock price forecasting and performance analysis.

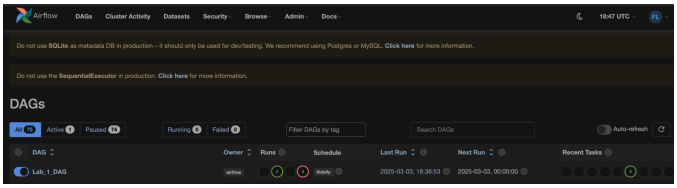

Fig. 6. Airflow UI showing ETL DAG created and manually triggered


Fig. 7. Airflow UI showing tasks in ETL DAG Graph

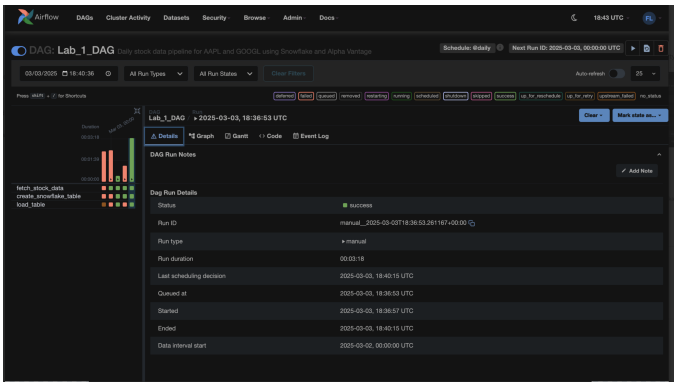


Fig. 8. Airflow UI of ETL DAG showing all 3 tasks ran successfully

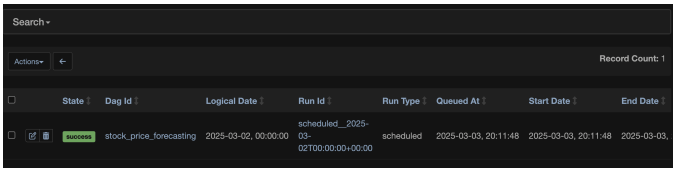


Fig. 9. Airflow UI showing tasks of Forecasting DAG

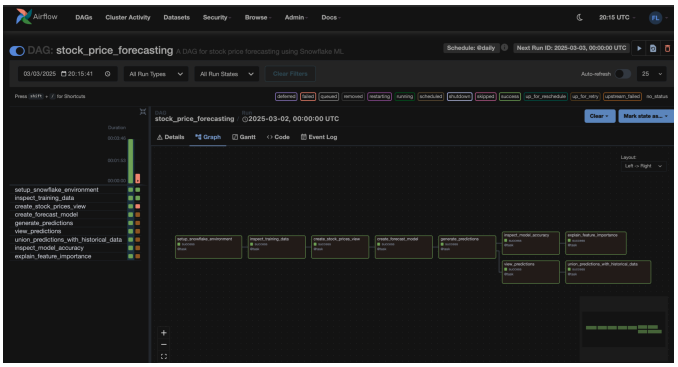


Fig. 10. Airflow UI showing Forecasting DAG Graph.

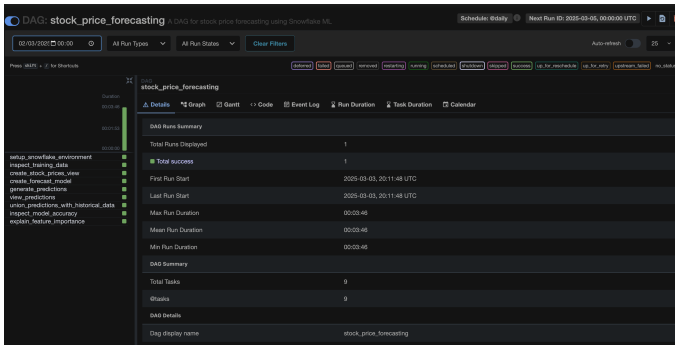


Fig. 11. Forecast DAG showing all the tasks completed successfully.

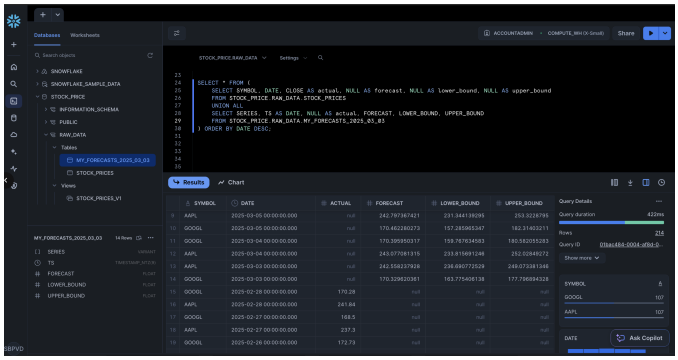


Fig. 12. Union of Forecast table and ETL table.

## V. External Links

The following link provides the Python codes and SQL queries used in building this machine learning model: **GitHub Repository Link**.

## References

[1] Yfinance: https://pypi.org/project/yfinance/
[2] Airflow: https://airflow.apache.org/docs/
[3] Snowflake: https://www.snowflake.com/en/data-cloud/workloads/ai-ml/