Stock Data Analytics Using Snowflake & Airflow

Xinxin Lin
San José State University
San Jose, California, USA 95192
xinxin.lin@sjsu.edu

Pragya Priyadarshini
San José State University
San Jose, California, USA 95192
pragya.priyadarshini@sjsu.edu

Abstract—**This report introduces a financial data analytics system that integrates Snowflake and Apache Airflow to automate the collection, storage, and analysis of stock market data. The system retrieves real-time stock information using the yfinance API, utilizes Snowflake as a data warehouse, and applies machine learning techniques for stock price forecasting. Apache Airflow manages the automation of daily data ingestion and updates. The findings highlight the effectiveness of an automated approach in financial analytics, enabling data-driven investment decisions through predictive analysis.**

Keywords—stock price prediction, ETL, data pipeline, Snowflake, Apache Airflow

I. Problem Statement

The stock market experiences significant fluctuations, making it essential for investors to have reliable tools for trend analysis and price forecasting. This project focuses on developing a Stock Price Prediction Analytics System by integrating Snowflake and Apache Airflow. The system will retrieve stock price data from the yfinance API, store it in Snowflake, and process it using machine learning techniques to generate forecasts. The model aims to predict stock prices for the next seven or more days based on the past 180 days of historical data.

Need for a Database and Data Pipelines: A database is essential for efficiently storing, managing, and analyzing historical stock price data. Given the continuous generation of stock market data, automated data pipelines play a key role in handling data ingestion, transformation, and machine learning model execution. This automation streamlines the forecasting process, ensuring consistency and accuracy without requiring manual intervention.

II. Solution Requirements

A. Functional Requirements

- Stock Data Extraction: Collect stock price metrics (Open, Close, High, Low, Volume) from the yfinance API.
- Data Storage: Organize and store the extracted data in a structured table within Snowflake.
- Automated Data Pipeline: Configure Apache Airflow DAGs to automate data ingestion and machine learning forecasting tasks.
- ML Model Training & Prediction: Train ARIMA or LSTM models to generate stock price forecasts for the next seven days.
- Result Integration: Combine historical stock data with forecasted values to create a comprehensive dataset.
- Error Handling: Implement SQL transactions and error-handling mechanisms to maintain system stability and data integrity.

B. Non-Functional Requirements

- Scalability: Designed to process large datasets efficiently while supporting multiple stock symbols.
- Reliability: Incorporates retry mechanisms and SQL transactions with try/except blocks to maintain data integrity.
- Security: Ensures secure storage of API keys and credentials using Airflow variables and connections.
- Performance: Optimizes SQL queries and data processing to enhance execution speed.
- Extensibility: Supports the integration of additional financial indicators and real-time market data for future enhancements.

C. System Limitations

- Model Accuracy: Forecasting is based on historical data and may not fully capture sudden market shifts.
- Data Source Dependency: The system depends on the yfinance API, which may experience data delays or temporary outages.
- Computational Cost: Processing large datasets with machine learning models demands substantial computational resources.

D. User Interaction

- Data Retrieval: Users can query the system to access stored stock data and forecasted results. Snowflake's querying interface enables seamless interaction with both raw and processed data.
- Automation Oversight: The data pipeline runs with minimal manual intervention, with Apache Airflow handling scheduling and execution efficiently.

III. Functional Analysis

A. System Components Overview

This system automates the full cycle of stock price data extraction, storage, processing, and forecasting. By integrating Snowflake as a data warehouse and Apache Airflow for workflow management, it ensures efficient automation and scalability. The system is structured into four key components:

1. Data Ingestion Layer: Retrieves real-time and historical stock data from the yfinance API.
2. Storage and Processing Layer: Organizes and processes stock data within Snowflake.
3. Forecasting and Analytics Layer: Applies machine learning models to predict future stock prices.
4. Automation and Orchestration Layer: Uses Apache Airflow DAGs to schedule and execute tasks.

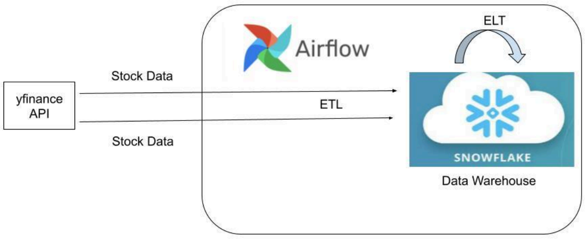


Fig 1: Data Pipeline: yfinance (Data) → Airflow → Snowflake (Analysis)

B. Data Acquisition: yfinance API

The yfinance API serves as the data source, providing stock market metrics such as open, high, low, close, and volume. The extracted data is structured into tables before being stored in Snowflake for further analysis.

C. Data Storage and Processing: Snowflake

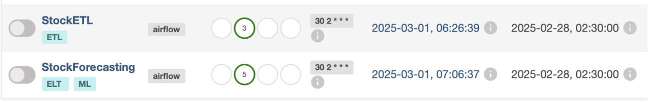
Stock market data within Snowflake is structured across three distinct schemas:

- Raw Data Schema: Stores unprocessed stock data retrieved from the yfinance API.
- Adhoc Schema: Used for intermediate transformations and machine learning analysis.
- Analytics Schema: Contains finalized results, including stock price predictions for visualization and reporting.

The ETL (Extract, Transform, Load) process ensures that data is properly cleaned, formatted, and loaded into Snowflake. SQL transactions and error-handling mechanisms are implemented to maintain consistency and reliability throughout the data pipeline.

D. Airflow Pipelines

Apache Airflow automates the execution of data pipelines by defining workflows as Directed Acyclic Graphs (DAGs). The system consists of two primary DAGs:



StockETL ETL	airflow	3	30 2 * * *	2025-03-01, 06:26:39	2025-02-28, 02:30:00
StockForecasting ELT ML	airflow	5	30 2 * * *	2025-03-01, 07:06:37	2025-02-28, 02:30:00

Fig 2: Directed Acyclic Graphs (DAGs)

1) ETL Pipeline (Extract, Transform, Load)

- *Task 1: Extract stock data from yfinance API.*
- *Task 2: Transform and clean 180 days' data.*
- *Task 3: Load transformed data into Snowflake.*

This pipeline ensures efficient handling of large datasets while applying validation rules before inserting data into Snowflake.

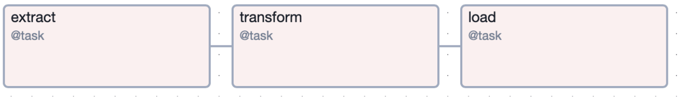


Fig 3: ETL Pipeline (Extract, Transform, Load)

2) ML Forecasting Pipeline (ELT Process)

- Task 1: Train the time-series forecasting model.
- Task 2: Predict stock prices for the next 7+ days.

Once stock data is stored in Snowflake, the forecasting pipeline can apply machine learning models to predict future stock prices. The system leverages Snowflake's ML capabilities to execute time-series forecasting models. The results are stored in the analytics schema, where they are merged with historical stock data to create a unified dataset for users.

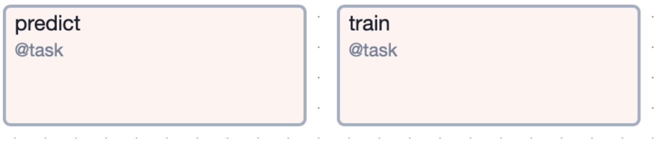


Fig 4: ML Forecasting Pipeline

E. Data Storage and Processing: Snowflake

1) Airflow Variables and Connections

- Variables: These handle API keys, stock tickers, and timeframes, allowing for dynamic management.
- Connections: These securely store credentials to facilitate connections to the Snowflake database.

2) Data Pipeline Execution

- Automated Scheduling: Airflow automates the scheduling of ETL and ML Forecasting pipelines at set intervals.
- Dependency Management: The ML Forecasting pipeline triggers only after the successful completion of the ETL pipeline.
- Retry Mechanisms: Set up to address issues with API failures and Snowflake connection errors.

3) Final Data Integration and Querying

- The final Snowflake table merges historical stock prices with predicted prices, enabling streamlined analysis.
- Users can query and visualize the data through SQL queries and BI tools like Tableau or Power BI.

IV. Table & Source Code

A. Tables Structure

TABLE I. Stock_Prices

Attribute	Data Type
SYMBOL	VARCHAR
DATE	DATE
OPEN	FLOAT
CLOSE	FLOAT
HIGH	FLOAT
LOW	FLOAT
VOLUME	NUMBER

TABLE II. Stock_Forecast

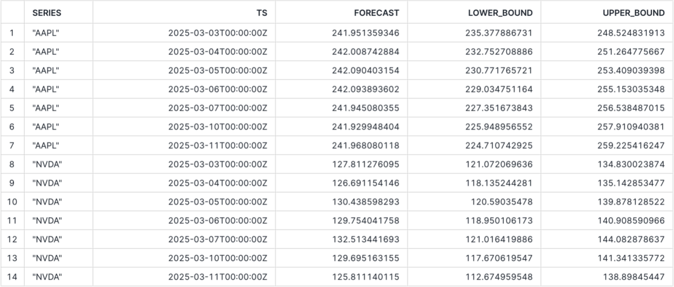
Attribute	Data Type
SERIES	VARIANT
DATE	DATE
TS	TIMESTAMP_NTZ
FORECAST	FLOAT
LOWER_BOUND	FLOAT
LOW	FLOAT
UPPER_BOUND	FLOAT

TABLE III. Stock_Price_Forecast

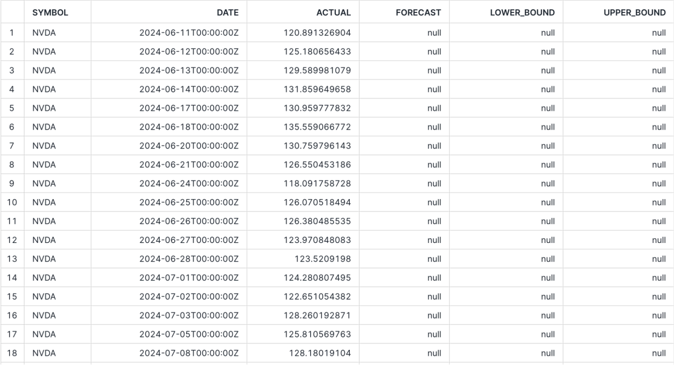
Attribute	Data Type
SYMBOL	VARCHAR
DATE	TIMESTAMP
ACTUAL	FLOAT
FORECAST	FLOAT
LOWER_BOUND	FLOAT
UPPER_BOUND	FLOAT

B. SQL Queries

Below are data previews of the Stock_Forecast table and Stock_Price_Forecast table (final table). To get the stock price predictions, the user will execute a SQL query in a Snowflake worksheet.

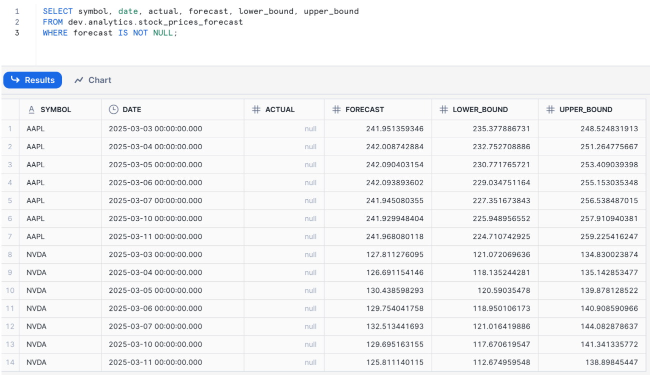


	SERIES	TS	FORECAST	LOWER_BOUND	UPPER_BOUND
1	"AAPL"	2025-03-03T00:00:00Z	241.951359346	235.377886731	248.524831913
2	"AAPL"	2025-03-04T00:00:00Z	242.008742884	232.752708886	251.264775667
3	"AAPL"	2025-03-05T00:00:00Z	242.090403154	230.771765721	253.409039398
4	"AAPL"	2025-03-06T00:00:00Z	242.093893602	229.034751164	255.153035348
5	"AAPL"	2025-03-07T00:00:00Z	241.945080355	227.351673843	256.538487015
6	"AAPL"	2025-03-10T00:00:00Z	241.929948404	225.948956552	257.910940381
7	"AAPL"	2025-03-11T00:00:00Z	241.968080118	224.710742925	259.225416247
8	"NVDA"	2025-03-03T00:00:00Z	127.811276095	121.072069636	134.830023874
9	"NVDA"	2025-03-04T00:00:00Z	126.691154146	118.135244281	135.142853477
10	"NVDA"	2025-03-05T00:00:00Z	130.438598293	120.59035478	139.878128522
11	"NVDA"	2025-03-06T00:00:00Z	129.754041758	118.950106173	140.908590966
12	"NVDA"	2025-03-07T00:00:00Z	132.513441693	121.016419886	144.082878637
13	"NVDA"	2025-03-10T00:00:00Z	129.695163155	117.670619547	141.341335772
14	"NVDA"	2025-03-11T00:00:00Z	125.811140115	112.674959548	138.89845447



	SYMBOL	DATE	ACTUAL	FORECAST	LOWER_BOUND	UPPER_BOUND
1	NVDA	2024-06-11T00:00:00Z	120.891326904	null	null	null
2	NVDA	2024-06-12T00:00:00Z	125.180656433	null	null	null
3	NVDA	2024-06-13T00:00:00Z	129.589981079	null	null	null
4	NVDA	2024-06-14T00:00:00Z	131.859649658	null	null	null
5	NVDA	2024-06-17T00:00:00Z	130.959777832	null	null	null
6	NVDA	2024-06-18T00:00:00Z	135.559066772	null	null	null
7	NVDA	2024-06-20T00:00:00Z	130.759796143	null	null	null
8	NVDA	2024-06-21T00:00:00Z	126.550453186	null	null	null
9	NVDA	2024-06-24T00:00:00Z	118.091758728	null	null	null
10	NVDA	2024-06-25T00:00:00Z	126.070518494	null	null	null
11	NVDA	2024-06-26T00:00:00Z	126.380485535	null	null	null
12	NVDA	2024-06-27T00:00:00Z	123.970848083	null	null	null
13	NVDA	2024-06-28T00:00:00Z	123.5209198	null	null	null
14	NVDA	2024-07-01T00:00:00Z	124.280807495	null	null	null
15	NVDA	2024-07-02T00:00:00Z	122.651054382	null	null	null
16	NVDA	2024-07-03T00:00:00Z	128.260192871	null	null	null
17	NVDA	2024-07-05T00:00:00Z	125.810569763	null	null	null
18	NVDA	2024-07-08T00:00:00Z	128.18019104	null	null	null

Fig 5: Data previews of Final table for AAPL & NVDA



```sql
SELECT symbol, date, actual, forecast, lower_bound, upper_bound
FROM dev.analytics.stock_prices_forecast
WHERE forecast IS NOT NULL;
```

	SYMBOL	DATE	ACTUAL	FORECAST	LOWER_BOUND	UPPER_BOUND
1	AAPL	2025-03-03 00:00:00.000	null	241.951359346	235.377886731	248.524831913
2	AAPL	2025-03-04 00:00:00.000	null	242.008742884	232.752708886	251.264775667
3	AAPL	2025-03-05 00:00:00.000	null	242.090403154	230.771765721	253.409039398
4	AAPL	2025-03-06 00:00:00.000	null	242.093893602	229.034751164	255.153035348
5	AAPL	2025-03-07 00:00:00.000	null	241.945080355	227.351673843	256.538487015
6	AAPL	2025-03-10 00:00:00.000	null	241.929948404	225.948956552	257.910940381
7	AAPL	2025-03-11 00:00:00.000	null	241.968080118	224.710742925	259.225416247
8	NVDA	2025-03-03 00:00:00.000	null	127.811276095	121.072069636	134.830023874
9	NVDA	2025-03-04 00:00:00.000	null	126.691154146	118.135244281	135.142853477
10	NVDA	2025-03-05 00:00:00.000	null	130.438598293	120.59035478	139.878128522
11	NVDA	2025-03-06 00:00:00.000	null	129.754041758	118.950106173	140.908590966
12	NVDA	2025-03-07 00:00:00.000	null	132.513441693	121.016419886	144.082878637
13	NVDA	2025-03-10 00:00:00.000	null	129.695163155	117.670619547	141.341335772
14	NVDA	2025-03-11 00:00:00.000	null	125.811140115	112.674959548	138.89845447

Fig 6: SQL Query Result for Forecast

C. Source Code

All SQL & Airflow Python codes are available in our GitHub Repo.