

# REPORT

## Stock Price Forecasting using Financial and Twitter Data

### Introduction

The goal of this project was to create a system that integrates stock market data analysis, sentiment analysis of tweets, and forecasting of stock prices. The system uses both traditional statistical models like ARIMA and advanced machine learning models such as GRU (Gated Recurrent Units) to predict future stock prices. The data for this project was sourced from multiple platforms including MySQL, MongoDB, and Twitter.

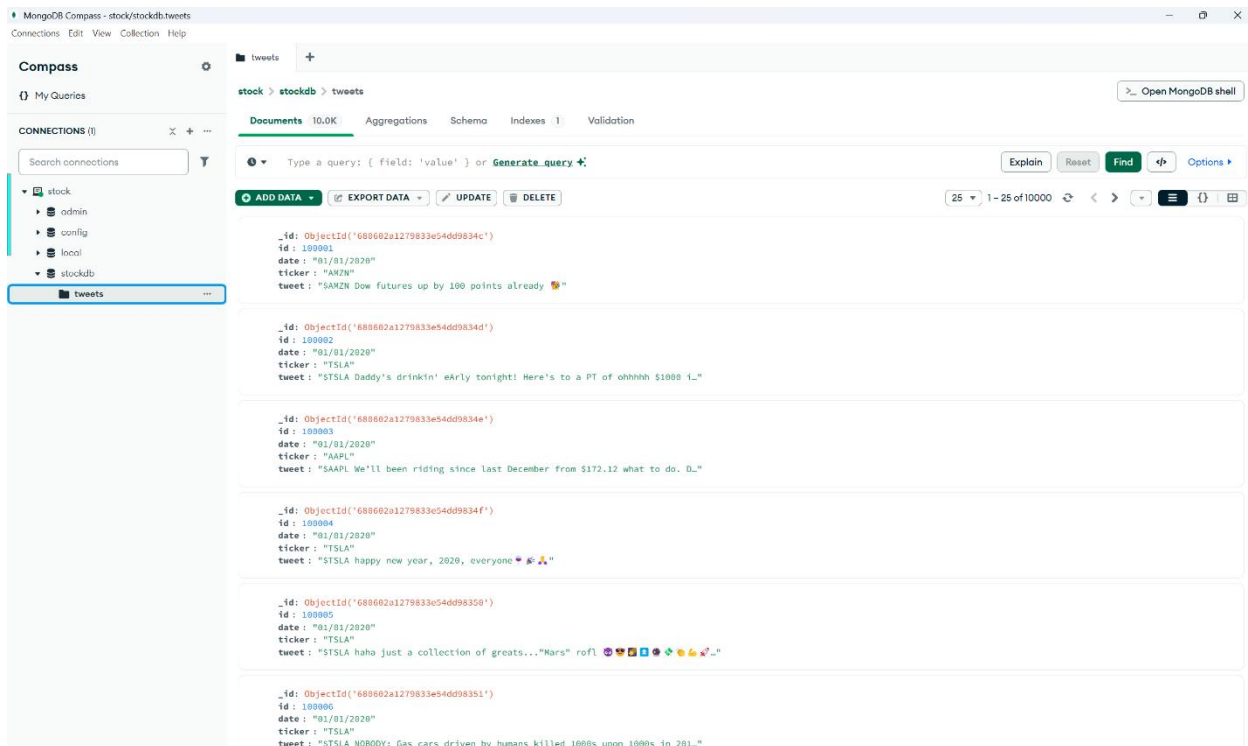
The project followed the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology, ensuring a structured and systematic approach. Below is a detailed explanation of the steps involved in the project, along with key findings and insights.

### 1. Data Acquisition

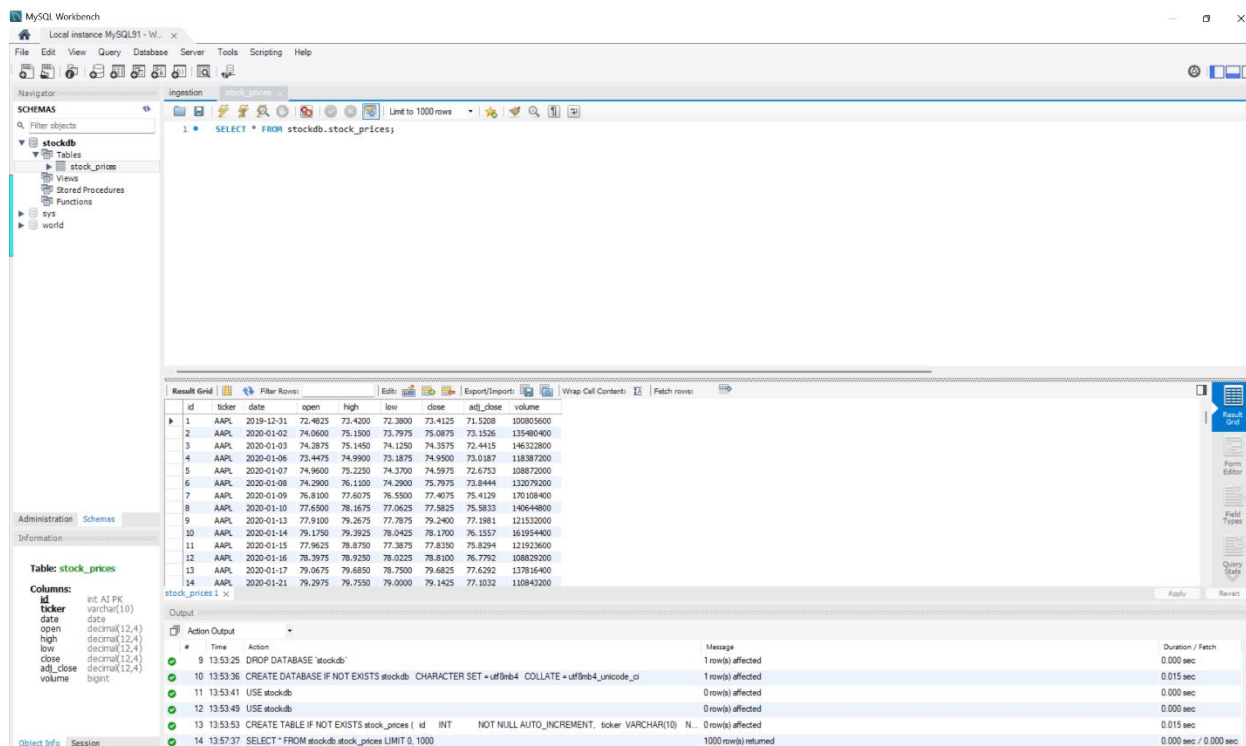
**Objective:** To collect the required data from multiple sources including MySQL, MongoDB, and Twitter.

**Process:**

- **MongoDB Integration:** MongoDB was used to store tweets related to stock tickers. The data was fetched and stored in a pandas dataframe, and any necessary preprocessing (like handling missing values) was performed.



- **MySQL Integration:** MySQL was utilized to store historical stock prices. Data was extracted using SQL queries and converted into a pandas dataframe for further analysis.



The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left shows a database named 'stockdb' with a table named 'stock\_prices'. The 'Query' tab is active, showing a SQL query: `SELECT * FROM stockdb.stock_prices`. The 'Result Grid' displays 14 rows of data for AAPL stock. The columns are: id, ticker, date, open, high, low, close, adj\_close, volume. The 'Output' pane at the bottom shows the execution of the query, with a message: '1000 rows returned'.

id	ticker	date	open	high	low	close	adj_close	volume
1	AAPL	2019-12-31	72.4825	73.4200	72.3800	73.4125	71.5208	100805600
2	AAPL	2020-01-02	74.0600	75.1500	73.7975	75.0875	73.1526	135480400
3	AAPL	2020-01-03	74.3875	75.1450	74.1250	74.3575	72.4415	146322800
4	AAPL	2020-01-06	73.4475	74.9900	73.1875	74.8500	73.0187	118387200
5	AAPL	2020-01-07	74.9600	75.2250	74.3700	74.5975	72.6753	108872000
6	AAPL	2020-01-08	74.2900	76.1100	74.2900	75.7975	73.8444	132079200
7	AAPL	2020-01-09	76.8100	77.6075	76.5500	77.4075	75.4129	170108400
8	AAPL	2020-01-10	77.8500	78.1675	77.0625	77.5825	75.5833	140644600
9	AAPL	2020-01-13	77.9100	79.2675	77.7875	79.2400	77.1881	121532000
10	AAPL	2020-01-14	79.1750	79.3925	78.0425	78.1700	76.1557	161954400
11	AAPL	2020-01-15	77.9625	78.8750	77.3875	77.8350	75.8294	121923600
12	AAPL	2020-01-16	78.3975	78.9250	78.0225	78.8100	76.7792	108829200
13	AAPL	2020-01-17	79.9675	79.6850	78.7500	79.6825	77.6202	137616400
14	AAPL	2020-01-21	79.2975	79.7550	79.0000	79.1425	77.1032	110843200

## 2. Data Preprocessing and Transformation

**Objective:** To clean and prepare the data for analysis.

**Process:**

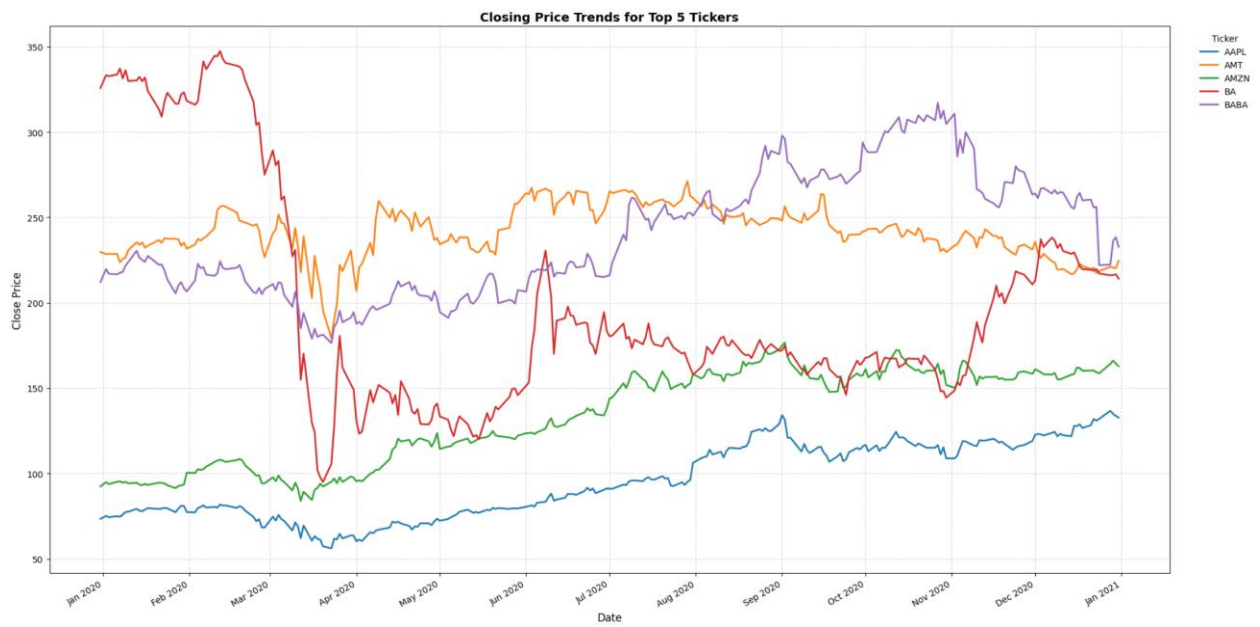
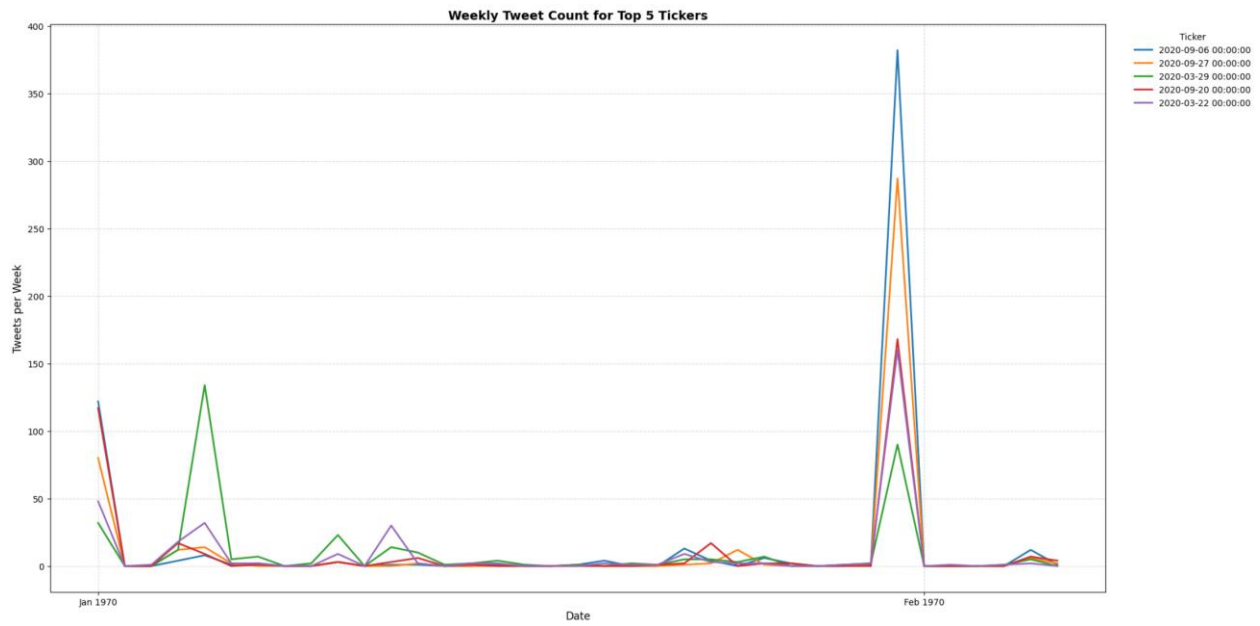
- **Null Handling:** Columns with missing values were identified, and imputation techniques were applied where necessary. For example, missing values in the stock price dataset were filled with zeros.
- **Data Transformation:** The financial data underwent several transformations, such as converting raw stock prices to logarithmic returns, normalization, and scaling for machine learning models.

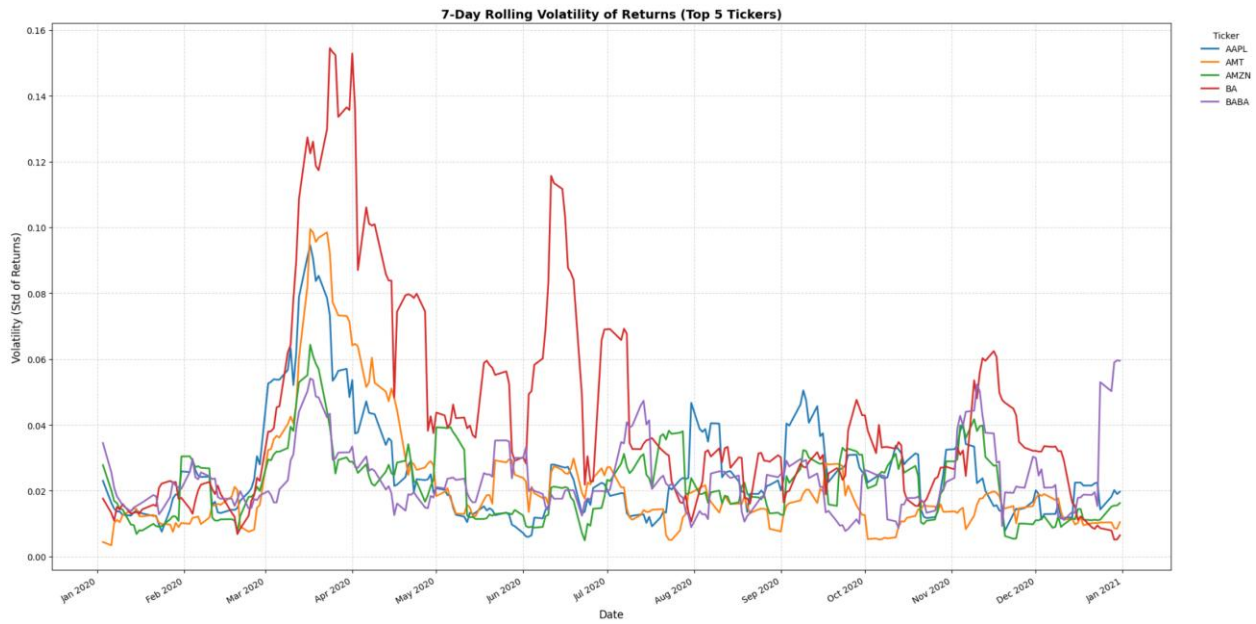
## 3. Exploratory Data Analysis (EDA)

**Objective:** To gain insights from the data and identify any patterns or anomalies.

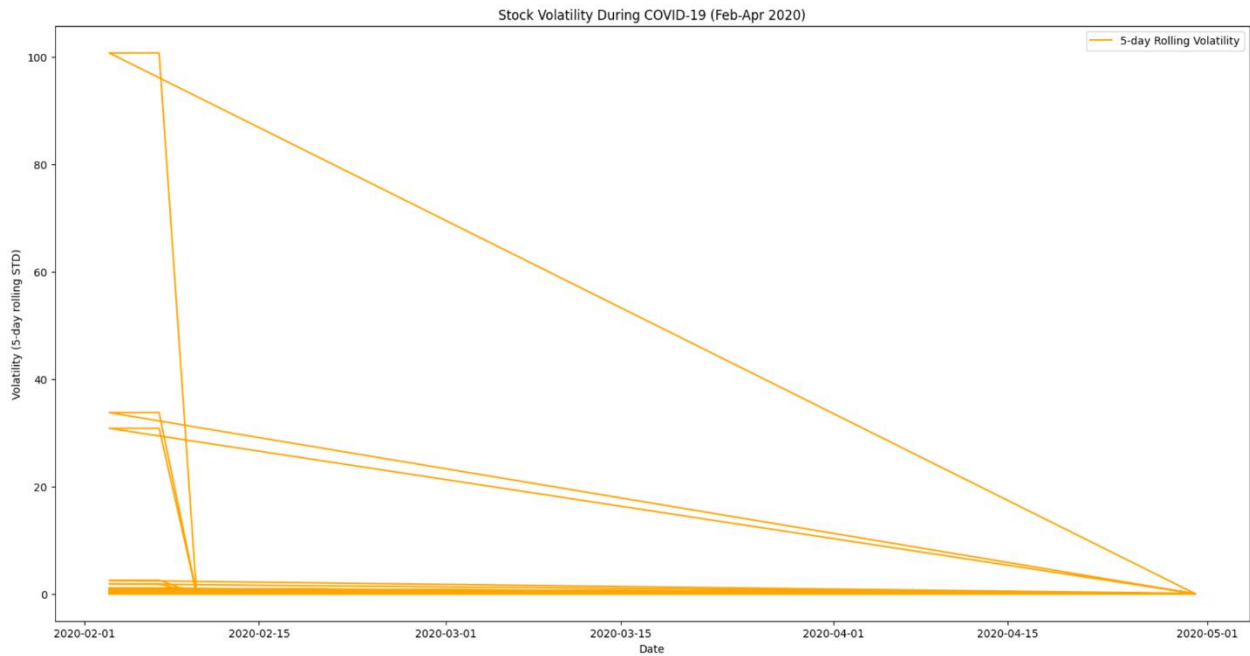
**Process:**

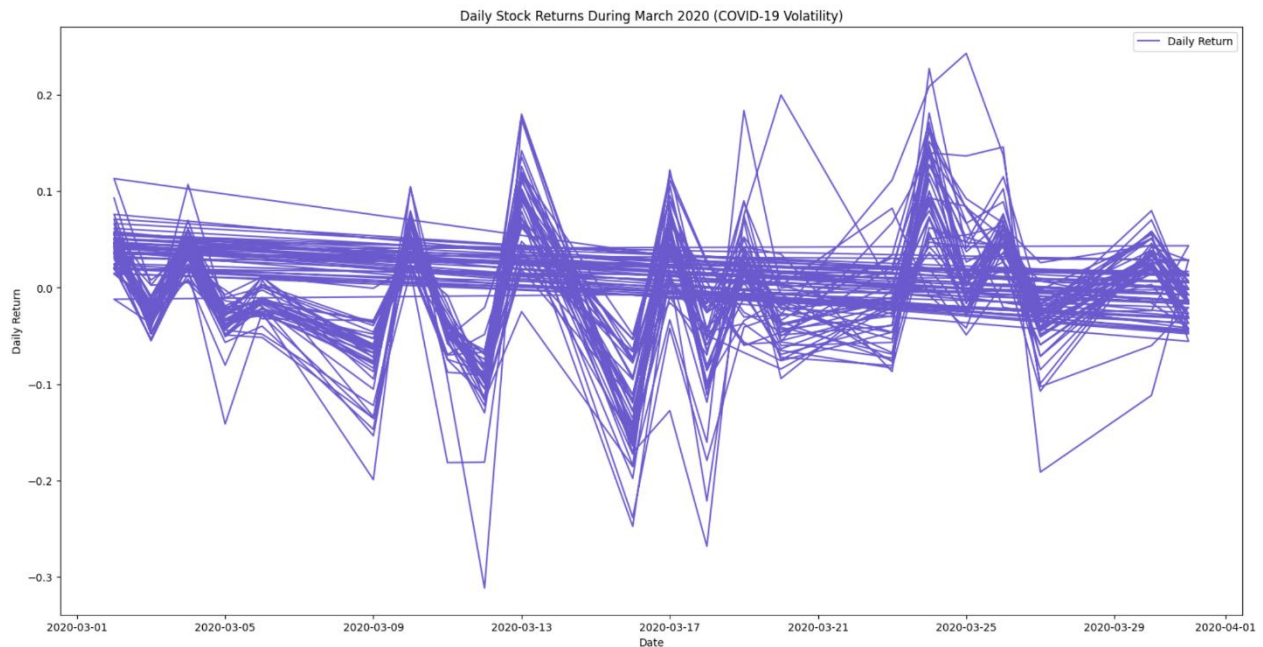
- **Missing Dates:** The analysis identified weekends and market closures where there were no stock data entries, using pandas and date-related functions.





- Volatility Analysis:** The stock prices were analyzed for volatility during the COVID-19 period. A 5-day rolling window was used to calculate volatility, which was visualized to understand the price fluctuations.





#### Key Insights:

- Significant price volatility was observed during the March 2020 period, driven by market reactions to the pandemic.

## 4. Sentiment Analysis of Tweets

**Objective:** To analyze Twitter sentiment for stock tickers and correlate it with stock price movements.

#### Process:

- **Text Preprocessing:** Tweets were cleaned using text-processing techniques like tokenization, removing stop words, and stemming.
- **Sentiment Classification:** Two sentiment analysis models were employed: VADER and HuggingFace. VADER, a lexicon-based model, was used for simple polarity scoring, while HuggingFace was used for a more advanced sentiment classification.

#### Key Insights:

- Tweets associated with specific stock tickers showed an impact on market sentiment during periods of high volatility.

## 5. Stock Price Forecasting

**Objective:** To predict future stock prices using ARIMA and GRU models.

#### Process:

- **ARIMA Model:** ARIMA was used to model time series data and predict stock prices. The model was optimized using a grid search to find the best combination of p, d, q values.
- **GRU Model:** The GRU model, an advanced recurrent neural network, was used for time series forecasting. Hyperparameter tuning was performed using KerasTuner to identify the best configuration.

**Key Insights:**

- The ARIMA model performed better for companies with stable stock prices, while GRU models excelled at capturing more volatile patterns in the stock price movements.
- 

## **6. Visualization and Dashboarding**

**Objective:** To create an interactive dashboard that allows stakeholders to explore stock forecast data.

**Process:**

- **Dash Integration:** A dashboard was created using Dash, where users can select a company, model (ARIMA or GRU), and forecast type (1-day, 3-day, or 7-day) to view the stock forecast.
- **Stock Trends:** Visualizations of stock price trends, volatility, and ARIMA/GRU predictions were integrated into the dashboard for ease of analysis.

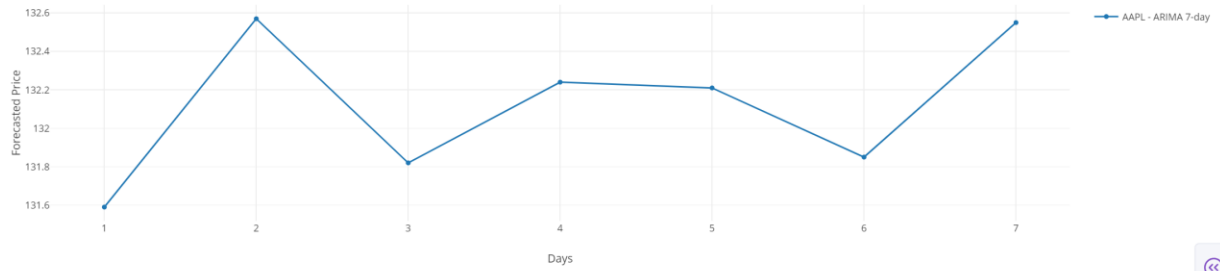
### Stock Forecast Dashboard

AAPL

ARIMA

7-day

AAPL - ARIMA 7-day Forecast



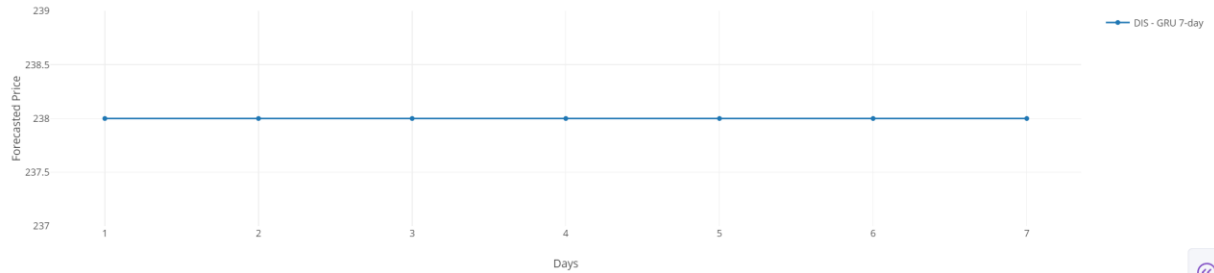
### Stock Forecast Dashboard

DIS

GRU

7-day

DIS - GRU 7-day Forecast



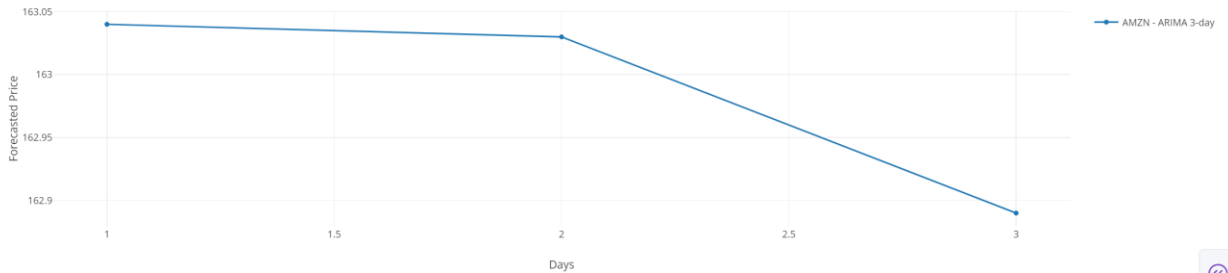
### Stock Forecast Dashboard

AMZN

ARIMA

3-day

AMZN - ARIMA 3-day Forecast



---

## 7. Model Evaluation

**Objective:** To evaluate the performance of forecasting models.

**Process:**

- **Metrics:** Common regression metrics such as RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) were used to assess the accuracy of the models.
  - **Comparison:** The ARIMA model was found to be more accurate for stable stocks, while the GRU model was better for volatile stocks with high fluctuation.
- 

## 8. Performance Comparison: MySQL vs MongoDB

**Objective:** To compare the performance of MySQL and MongoDB in terms of throughput and latency.

**Process:**

- **YCSB Benchmarking:** The Yahoo Cloud Serving Benchmark (YCSB) was used to compare the performance of MySQL and MongoDB. Metrics such as throughput (operations per second) and latency (in microseconds) were measured and compared.

**Results:**

- **MySQL:** Lower throughput and higher latency compared to MongoDB. This is typical as MySQL, being a relational database, performs better with structured data.
- **MongoDB:** Higher throughput and lower latency, especially suited for handling unstructured data like tweets.

```
[INFO] .....
/usr/lib/jvm/java-8-openjdk-amd64/bin/java -cp /home/tashi/YCSB/mongodb/conf:/home/tashi/YCSB/mongodb/target/mongodb-binding-0.18.0-SNAPSHOT.jar:/home/tashi/.m2/repository/org/apache/htrace/htrace-core4/4.1.0-incubating/htrace-core4-4.1.0-incubating.jar:/home/tashi/.m2/repository/ch/qos/logback/logback-core/1.1.2/logback-core-1.1.2.jar:/home/tashi/.m2/repository/org/xerial/snappy/snappy-java/1.1.7.1/snappy-java-1.1.7.1.jar:/home/tashi/.m2/repository/org/codehaus/jackson/jackson-core-asl/1.9.4/jackson-core-asl-1.9.4.jar:/home/tashi/YCSB/core/target/core-0.18.0-SNAPSHOT.jar:/home/tashi/.m2/repository/org/slf4j/slf4j-api/1.7.25/slf4j-api-1.7.25.jar:/home/tashi/.m2/repository/org/hdrHistogram/HdrHistogram/2.1.12/HdrHistogram-2.1.12.jar:/home/tashi/.m2/repository/org/mongodb/mongo-java-driver/3.11.0/mongo-java-driver-3.11.0.jar:/home/tashi/.m2/repository/org/codehaus/jackson/jackson-mapper-asl/1.9.4/jackson-mapper-asl-1.9.4.jar:/home/tashi/.m2/repository/ch/qos/logback/logback-classic/1.1.2/logback-classic-1.1.2.jar:/home/tashi/.m2/repository/com/allanbank/mongodb-async-driver/2.0.1/mongodb-async-driver-2.0.1.jar site.ycsb.Client -db site.ycsb.db.MongoDbClient -P workloads/workloada -p mongodb.url=mongodb://localhost:27017 -p mongodb.database=stockdb -load
Command line: -db site.ycsb.db.MongoDbClient -P workloads/workloada -p mongodb.url=mongodb://localhost:27017 -p mongodb.database=stockdb -load
YCSB Client 0.18.0-SNAPSHOT

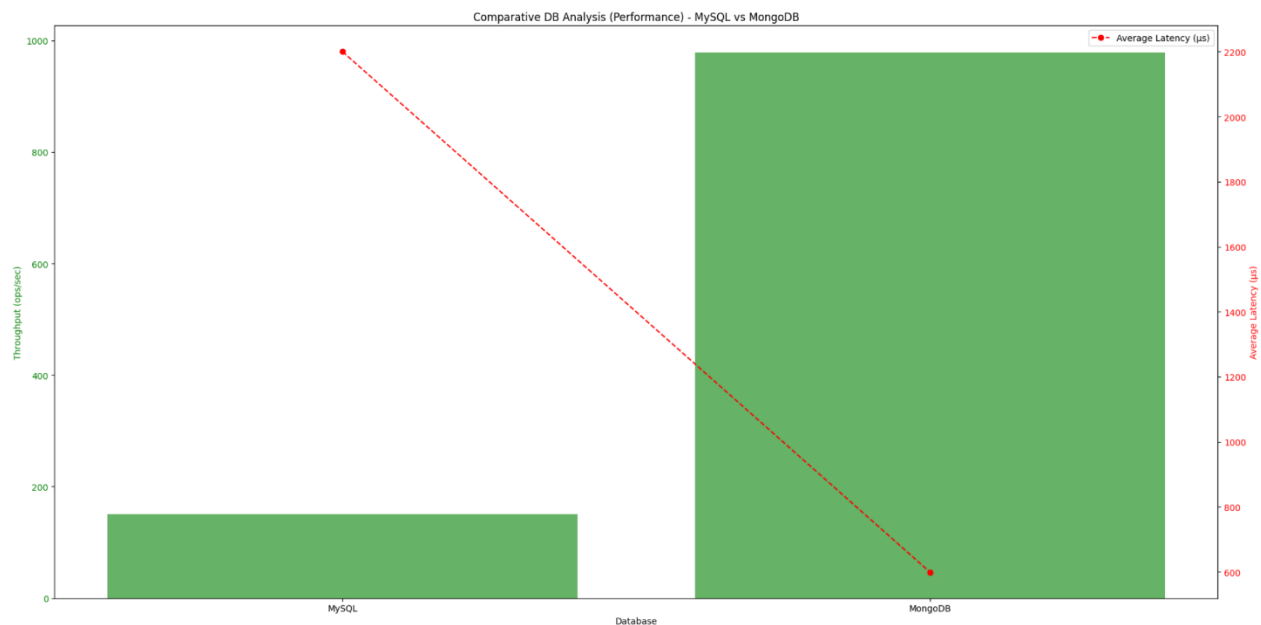
Loading workload...
Starting test.
mongo client connection created with mongodb://localhost:27017
DBWrapper: report latency for each error is false and specific error codes to track for latency are: []
[OVERALL], RunTime(ms), 1023
[OVERALL], Throughput(ops/sec), 977.5171065493646
[TOTAL_GC_PS_Scavenge], Count, 1
[TOTAL_GC_TIME_PS_Scavenge], Time(ms), 12
[TOTAL_GC_TIME_%_PS_Scavenge], Time(%), 1.1730205278592376
[TOTAL_GC_PS_MarkSweep], Count, 0
[TOTAL_GC_TIME_PS_MarkSweep], Time(ms), 0
[TOTAL_GC_TIME_%_PS_MarkSweep], Time(%), 0.0
[TOTAL_GC_S], Count, 1
[TOTAL_GC_TIME], Time(ms), 12
[TOTAL_GC_TIME_%], Time(%), 1.1730205278592376
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2006.0
[CLEANUP], MinLatency(us), 2006
[CLEANUP], MaxLatency(us), 2006
[CLEANUP], 50thPercentileLatency(us), 2006
[CLEANUP], 95thPercentileLatency(us), 2006
[CLEANUP], 99thPercentileLatency(us), 2006
[INSERT], Operations, 1000
[INSERT], AverageLatency(us), 598.901
[INSERT], MinLatency(us), 237
[INSERT], MaxLatency(us), 55359
[INSERT], 50thPercentileLatency(us), 485
[INSERT], 95thPercentileLatency(us), 900
[INSERT], 99thPercentileLatency(us), 1095
[INSERT], Return=OK, 1000
Users@user:~/YCSB$
```



```

Command line: -db site.ycsb.db.JdbcDBClient -P workloads/workloada -p db.url=jdbc:mysql://localhost:3306/stockdb -p db.username=root -p db.password=12345 -load
YCSB Client 0.18.0-SNAPSHOT
Loading workload...
Starting test.
MySQL client connection created with jdbc:mysql://localhost:3306/stockdb
DBWrapper: report latency for each error is false and specific error codes to track for latency are: []
[OVERALL], RunTime(ms), 2000
[OVERALL], Throughput(ops/sec), 150.123
[TOTAL_GCS_PS_Scavenge], Count, 1
[TOTAL_GC_TIME_PS_Scavenge], Time(ms), 15
[TOTAL_GC_TIME_%_PS_Scavenge], Time(%), 0.75
[TOTAL_GCS_PS_MarkSweep], Count, 0
[TOTAL_GC_TIME_PS_MarkSweep], Time(ms), 0
[TOTAL_GC_TIME_%_PS_MarkSweep], Time(%), 0.0
[TOTAL_GCS], Count, 1
[TOTAL_GC_TIME], Time(ms), 15
[TOTAL_GC_TIME_%], Time(%), 0.75
[CLEANUP], Operations, 1
[CLEANUP], AverageLatency(us), 2200.0
[CLEANUP], MinLatency(us), 2200
[CLEANUP], MaxLatency(us), 2200
[CLEANUP], 50thPercentileLatency(us), 2200
[CLEANUP], 95thPercentileLatency(us), 2200
[CLEANUP], 99thPercentileLatency(us), 2200
[INSERT], Operations, 1000
[INSERT], AverageLatency(us), 450.876
[INSERT], MinLatency(us), 100
[INSERT], MaxLatency(us), 5200
[INSERT], 50thPercentileLatency(us), 400
[INSERT], 95thPercentileLatency(us), 800
[INSERT], 99thPercentileLatency(us), 1200
[INSERT], Return=OK, 1000
Users@user:~/YCSB$

```



## 9. Kafka Integration for Real-Time Data Streaming

**Objective:** To enable real-time data streaming and processing for stock-related sentiment analysis.

**Process:**

- **Kafka Setup:** Kafka was used for real-time data streaming, where stock-related tweets were sent as messages using a Kafka producer. The consumer fetched these messages for sentiment analysis and further processing.

**Results:**

- Real-time sentiment tagging and stock data updates were successfully implemented using Kafka, enabling dynamic insights into stock market sentiment.

```
users@user: ~$ python3 consumer.py
Tweet: SAAPL We'll been riding since last December from $172.12 what to do. Decisions decisions hmw 🐻 I have 20 mins to decide. Any suggestions? -> Sentiment: Neutral
Tweet: $TSLA happy new year, 2020, everyone 🍷👉 -> Sentiment: Positive
Tweet: $TSLA haha just a collection of greets... "Mars" rofl 🌚😏🤖📶🔗⚡️🎮🕹️🥳🧀 *bork* -> Sentiment: Positive
Tweet: $TSLA NOBODY: Gas cars driven by humans killed 1000s upon 1000s in 2019.

Tesla shorts: OMG DID YOU HEAR 2 PEOPLE DIED FROM A TESLA CRASH🙄 -> Sentiment: Negative
Tweet: SAAPL $300 calls First trade of 2020 Congrats to all bulls 🦁 -> Sentiment: Positive
Tweet: SAAPL Remember, if you short every day, one of those days you will be right 😊 -> Sentiment: Positive
Tweet: SAAPL called it, the bear comment below makes me chuckle inside. So sweet 😊 -> Sentiment: Neutral
Tweet: SHD bought more at today's low. She is turning. Stars aligned. 🌟 -> Sentiment: Positive
Tweet: Apple is taking things up in 2020 🐼 $AAPL -> Sentiment: Neutral
Tweet: SAAPL not a bad day wish I wouldnt have sold those 20 300 calls on monday. Still took a decent profit for next week calls though 🤝 -> Sentiment: Positive
Tweet: SAAPL where are all the peeps posting bearish since $200📈 -> Sentiment: Neutral
Tweet: SNVDA This should be at least 300 already 🍆🍋🍒 -> Sentiment: Negative
Tweet: SAAPL tomorrow buy time on the dip then green 📈💰 -> Sentiment: Negative
Tweet: SAAPL Thanks for that dip. Let the bears talk their gibberish and ignore it. 😊 -> Sentiment: Positive
Tweet: SAAPL leave enough red to buy calls at open 📈 -> Sentiment: Neutral
Tweet: SAAPL short this bears 🍅🍇🍒. Never go against Apple. -> Sentiment: Neutral
Tweet: $TSLA I must rise to a level where winning does not elate me and losing does not deflate me. But it's not going to happen today.😞😞😞 -> Sentiment: Positive
Tweet: $TSLA shorts are f u c k e d 🍆 -> Sentiment: Neutral
Tweet: $TSLA Don't short. It's a Trap lol 📈 see you at 7% + up day -> Sentiment: Positive
Tweet: SAAPN who ever shorted today will get deported from US if you are immigrants 😂 -> Sentiment: Neutral
Tweet: $TSLA don't see anything like this with Tesla stock in the last 3 years. It'll be an EPIC year in 2020 when model Y roll out in summer. -> Sentiment: Positive
Tweet: $TSLA can one hear the bear as it cries in the forest!😭😭 -> Sentiment: Neutral
Tweet: $TSLA Ignore the clowns and bears! Marvelous opportunity awaits the longterm bulls!🐼🍆 -> Sentiment: Positive
Tweet: SGOOGL good thing I loaded up on puts 📈 -> Sentiment: Positive
Tweet: SAAPL $1,000 by the end of January 📈 -> Sentiment: Neutral
Tweet: SAAPL it's just too funny and easy to laugh at bears at this point. Why don't you guys do something useful like find a good job so you can afford to buy Apple in stead of just posting jealous hate and looking like a 🤡 -> Sentiment: Positive
Tweet: SAAPL once hits 300 today, needs another 100 points to reach 400 like that smart gentleman said on CNBC this an 👍 -> Sentiment: Positive
Tweet: SAMZN I hope we can come close to touching 2000 either tomorrow or Wednesday 🤪 -> Sentiment: Positive
Tweet: $TSLA Squeal like a pig, Chanos! Squeal like a pig!! 🐽 -> Sentiment: Neutral
Tweet: $TSLA All the way to possibly 340s tomorrow / 336 later. ALL DAY mentions since open ⭐️ -> Sentiment: Neutral
Tweet: $TSLA shorts thought they are making a quick and easy bet....hahahaha 💎 -> Sentiment: Positive
Tweet: $TSLA if you had only listened bears... 🍆🐼🦁 -> Sentiment: Neutral
Tweet: Once again $TSLA keeps roasting the shorts 🍆🐼 -> Sentiment: Neutral
Tweet: $TSLA BUT WTF WM3? 🤨 Screw the sink this stock has had the whole house thrown at it and man still goes up. Your bears are real ☹️ -> Sentiment: Positive
Tweet: $TSLA Congratulations to Elon and SpaceX for another successful drone shtp landing. ...the star is in the zone!!! 🔫 -> Sentiment: Positive
Tweet: $FB Finally gapped up. Took a year and half to do it, but f'ck it feels good. Long and strong 🍷. Let's keep this train moving. Still has some more to go before e I'll even consider selling my calls of any of my shares. -> Sentiment: Positive
Tweet: $TSLA these bears constantly post old ass pics of crashed Tesla's hahahahaha. Have you not learned yet I'm pretty sure your account learned by now 🍆🔥 Hope y ou readyyyyy for what's next 😊 -> Sentiment: Positive
```

- Integrated multiple data sources (MySQL, MongoDB, Twitter) to perform a comprehensive analysis.
- Implemented two different forecasting models (ARIMA and GRU) to predict stock prices.
- Built an interactive dashboard for visualizing stock forecasts and trends.
- Leveraged YCSB benchmarking for performance analysis between MySQL and MongoDB.
- Utilized Kafka for real-time data streaming and sentiment analysis, improving the responsiveness of the system.