**Comparative Analysis of MySQL and MongoDB for Stock Price Prediction and Sentiment Analysis**

Table of Contents

[Introduction 5](#_Toc184496632)

[Objectives of the Project 5](#_Toc184496633)

[Overview of Datasets and Tools: 5](#_Toc184496634)

[Significance of the Study: 6](#_Toc184496635)

[Big Data Storage and Processing 6](#_Toc184496636)

[Data Storage in MySQL 6](#_Toc184496637)

[Advantages for MySQL regarding this project: 6](#_Toc184496638)

[MongoDB Storage 7](#_Toc184496639)

[Advantages of Using MongoDB for This Project 7](#_Toc184496640)

[Comparative Analysis of MySQL vs MongoDB 7](#_Toc184496641)

[Data Ingestion Time: 7](#_Toc184496642)

[Scalability: 7](#_Toc184496643)

[Comparison-based Insights 8](#_Toc184496644)

[MySQL Usage Cases: 8](#_Toc184496645)

[Usage Cases for MongoDB: 8](#_Toc184496646)

[Data Preprocessing 9](#_Toc184496647)

[Steps for Cleaning and Preparing Data 9](#_Toc184496648)

[Tweet Cleaning: 9](#_Toc184496649)

[Stock Price Data Cleaning: 9](#_Toc184496650)

[Dataset Merging 9](#_Toc184496651)

[Challenges and Resolutions 9](#_Toc184496652)

[Handling Missing Data: 9](#_Toc184496653)

[Memory Constraints 10](#_Toc184496654)

[Inconsistent Date Formats 10](#_Toc184496655)

[Sentiment Analysis 10](#_Toc184496656)

[Techniques Used for Sentiment Analysis 10](#_Toc184496657)

[TextBlob: 10](#_Toc184496658)

[VADER: Valence Aware Dictionary and sEntiment Reasoner. 10](#_Toc184496659)

[Techniques: 10](#_Toc184496660)

[Benefits: 11](#_Toc184496661)

[Drawbacks 11](#_Toc184496662)

[Implementation 11](#_Toc184496663)

[Preprocessing Tweets: 11](#_Toc184496664)

[Sentiment Scoring: 11](#_Toc184496665)

[Aggregating Sentiment: 11](#_Toc184496666)

[Visualization of Sentiment Trends 11](#_Toc184496667)

[Bar Charts: 11](#_Toc184496668)

[Heatmaps: 11](#_Toc184496669)

[Results 11](#_Toc184496670)

[Challenges and Solutions 12](#_Toc184496671)

[Time-Series Forecasting 12](#_Toc184496672)

[ARIMA Technique 12](#_Toc184496673)

[Data Preprocessing: 12](#_Toc184496674)

[Modelling: 12](#_Toc184496675)

[ Parameters (p, d, q) were optimized using the Akaike Information Criterion (AIC). 12](#_Toc184496676)

[ ARIMA was trained on historical stock prices for each company. 12](#_Toc184496677)

[Fore Cast: 12](#_Toc184496678)

[Performance: 12](#_Toc184496679)

[LSTM Approach 12](#_Toc184496680)

[Data Preprocessing: 12](#_Toc184496681)

[Model Development: 12](#_Toc184496682)

[Training and Hyperparameter Tuning: 13](#_Toc184496683)

[Performance: 13](#_Toc184496684)

[Performance Comparison 13](#_Toc184496685)

[Evaluation Metrics: 13](#_Toc184496686)

[Findings and Insights 13](#_Toc184496687)

[Challenges 13](#_Toc184496688)

[Low Data: 13](#_Toc184496689)

[Hyperparameter Tuning: 13](#_Toc184496690)

[Interactive Dashboard 13](#_Toc184496691)

[Features and Functionalities 14](#_Toc184496692)

[Sentiment Analysis Visualization: 14](#_Toc184496693)

[Forecasting Visualizations 14](#_Toc184496694)

[Interactive filters 14](#_Toc184496695)

[Performance metrics: 14](#_Toc184496696)

[Design Principles (Tufts) 14](#_Toc184496697)

[Clarity: 14](#_Toc184496698)

[Accessibility: 14](#_Toc184496699)

[Aesthetics: 14](#_Toc184496700)

[Technology Used 15](#_Toc184496701)

[Challenges and Solutions 15](#_Toc184496702)

[Dynamic Filtering: 15](#_Toc184496703)

[User Interaction Design 15](#_Toc184496704)

[Conclusion and Recommendations 15](#_Toc184496705)

[Conclusion 15](#_Toc184496706)

[Recommendations 16](#_Toc184496707)

[Dataset Expansion: 16](#_Toc184496708)

[Integrate Advanced NLP Models: 16](#_Toc184496709)

[Real-Time Data Integration: 16](#_Toc184496710)

[Hybrid Forecasting Models: 16](#_Toc184496711)

[Scalability Testing: 16](#_Toc184496712)

[User-Centric Dashboard Enhancements: 16](#_Toc184496713)

[Final Thoughts 16](#_Toc184496714)

[References 17](#_Toc184496715)

# Introduction

In the digital era, the exponential growth of data necessitates efficient storage, processing, and analytical techniques to derive actionable insights. This project aims to explore the comparative advantages of SQL (MySQL) and NoSQL (MongoDB) databases for storing and processing big data, focusing on their application in financial forecasting and sentiment analysis. This study investigates the impact of market sentiment on stock trends by leveraging data from Twitter and historical stock prices and uses advanced forecasting models to predict stock prices (ALI et al., 2023).

## Objectives of the Project

This project mainly focuses on the performance of MySQL and MongoDB in terms of data ingestion, querying, and scalability. The study further integrates sentiment analysis to classify the tweets about selected companies into positive, neutral, or negative sentiments and analyse the impact of these sentiments on market movements. Advanced time-series forecasting methods, including ARIMA and LSTM, are implemented to predict the stock prices for 1, 3, and 7 days ahead and provide actionable insights for investors and financial analysts (Khan et al., 2023).

## Overview of Datasets and Tools:

**The datasets used in this project include:**

**stocktweet.csv:** A dataset of 10,000 tweets from January to December 2020, including the text of the tweet, the company's ticker symbol, and the date posted (Mavrogiorgos et al., 2021).

**Stock Price Data:** Historical prices of 38 companies over the period, including fields such as Date, Open, Close, and Volume.

**These are some of the tools used:**

**MySQL:** Relational database system to store and process structured data.

**MongoDB :** NoSQL database designed for handling unstructured and semi-structured data.

**Python:** pandas, scikit-learn, tensorflow, and plotly dash would be the main libraries related to data analysis, machine learning, and visualization.

**Apache Spark:** Scalable and High-Performance Distributed Data Processing (Győrödi et al., 2022).

## Significance of the Study:

This project addresses the challenge of handling and analysing large volumes of heterogeneous data in the financial domain. It offers insights into the selection of the right database based on use-case requirements by comparing MySQL and MongoDB (Matallah, Belalem and Bouamrane, 2021). The integration of sentiment analysis and forecasting models further demonstrates how data-driven techniques can support informed financial decision-making. Results from this study should, therefore, benefit data engineers, financial analysts, and researchers working at the interface of technology and finance (Rao et al., 2022).

# Big Data Storage and Processing

Efficient storage and processing are critical in modern analytics projects and especially when dealing with high volumes of structured and unstructured data. This chapter studies the storage methods and MySQL (SQL database) vs. MongoDB (NoSQL database) performance while keeping tweet and stock price data in mind. Both databases are compared in terms of data ingestion, query performance and scalability to provide important comparison insights for big data (van der Veen, van der Waaij and Meijer, 2012).

## Data Storage in MySQL

A widely available relational database management system, MySQL, was selected for its structured approach and the adherence to ACID (Atomicity, Consistency, Isolation, Durability) properties. The dataset was split into two tables:

**Tweets Table:**

* **Columns:** id, date, ticker, tweet.
* **Primary key:** id to maintain uniqueness for easy querying.

**Stock Prices Table:**

* **Columns:** date, ticker, open, high, low, close, adj\_close, volume.
* Indexed by date and ticker for efficient filtering.

## Advantages for MySQL regarding this project:

* **Schema Validation:** The schema also provides predefined validation against inconsistencies and redundancy.
* **Relational Queries:** MySQL is great for merging the tweet and stock price datasets using date and ticker due to support for complex joins.
* **Security and Reliability:** Built-in authentication and transactional features ensure that data integrity is critical in financial data.

## MongoDB Storage

MongoDB is used due to its schema-less design and the ability to handle unstructured data. The two collections used are:

**Tweets Collection:**

* Documents will include id, date, ticker, and tweet.
* Text data, which is the tweet, will be stored as part of each document.

**Stock Prices Collection:**

* Each document will include date, ticker, and price-related fields.
* Flexible schema allows changes as required with the data that will eventually come without changing the schema.

## Advantages of Using MongoDB for This Project

* Flexibility: MongoDB is great with the unstructured data of tweets as they may come in many different types.
* Scalability: Sharding provides horizontal scaling to deal with big sizes of data sets.
* High-Speed Ingestion: Batch document insertion is quicker than SQL row-based insertion.

## Comparative Analysis of MySQL vs MongoDB

The databases were compared across three key metrics: data ingestion time, query performance, and scalability. Benchmarking tests were performed using sample datasets and real queries concerning sentiment analysis and stock price forecasting (Shah, Jat and Sashidhar, 2022).

### Data Ingestion Time:

MongoDB outperformed MySQL because it can insert the documents without schema validation overhead.

* Ingestion of 10,000 tweets:
* MySQL: 4.256014585494995 seconds.
* MongoDB: 0.1536686420440673 seconds.

MySQL was faster for the queries that involved joins on structured data, such as combining the tweets and stock prices by date and ticker.

* Time for calculating the average closing price of a stock
* MySQL: 0.02513432502746582 seconds.
* Mongo DB: 0.09668684005737305 seconds.

### Scalability:

* MongoDB scaled better as the dataset size increased. Its architecture supports distributed architecture.
* MySQL performance started slowing down with the increase in data due to its monolithic structure.

## Comparison-based Insights

### MySQL Usage Cases:

* Ideal for applications requiring strict data integrity and structured queries.
* Best for financial data merges and analysis where relational operations are predominant.

### Usage Cases for MongoDB:

* Highly suitable for handling unstructured data at high volume such as tweets.
* Most suited for real-time analytics or applications that require high-speed ingestion and flexible updates to schema.

# Data Preprocessing

Preprocessing of data is an important step for any data analytics project so that the quality and consistency of data are guaranteed for the analysis. The project worked with two datasets: the tweets from stocktweet.csv and the stock price of 38 companies in separate CSV files. These datasets were pre-processed to clean, merge, and prepare them for sentiment analysis and time-series forecasting (Wang et al., 2024).

## Steps for Cleaning and Preparing Data

### Tweet Cleaning:

Tweet cleaning involved the extraction of columns id, date, ticker, and tweet from the stocktweet.csv file.

**Operations included:**

* **Removing Special Characters and URLs:** Text cleaning was carried out to eliminate all non-alphanumeric characters, hyperlinks, and white spaces by using the Python's re library
* **Case Normalization:** All texts were made in lower cases to reduce redundancy in the analytical process.
* **Date Normalization:** The date column was formatted in datetime for uniformity across the datasets (Malebary and Abulfaraj, 2024).

### Stock Price Data Cleaning:

* Stock price files- all columns Date, Open, High, Low and Close were cleaned. All missing entries, which predominantly occurred in the Volume column, were filled using the median value for the corresponding stock. Extreme outliers that appeared in the price columns were dealt with using statistical methods like interquartile range, IQR. The Date column is also converted to datetime.

### Dataset Merging

* The stock price data was merged with the tweets dataset using the date and ticker columns. This was done via an inner join to make sure only common entries are considered.
* Further features calculated and appended to the merged dataset include average daily sentiment and price movement indicators.

## Challenges and Resolutions

### Handling Missing Data:

* **Problem:** The stock price data included missing entries due to holidays or incomplete trading sessions.
* **Solution:** Missing dates were interpolated linearly for numerical columns such as Close and forward-filled for categorical columns like Ticker.

### Memory Constraints

* **Problem:** Merging massive datasets led to performance-related problems on standard systems.
* **Solution:** Data was processed in chunks using Python's dask library, which allowed dealing with large files efficiently

### Inconsistent Date Formats

* **Challenge:** The date formats differed for each dataset.
* **Resolution:** Their date variables are standardized into a single date format by using pandas.to\_datetime.

# Sentiment Analysis

Sentiment analysis is a technique of NLP used to analyse the mood or opinion expressed in textual data. In this project, sentiment analysis was applied to the tweets from the stocktweet.csv dataset to determine their emotional polarity, whether positive, neutral, or negative. The results were used to identify market sentiment and assess its influence on stock prices.

## Techniques Used for Sentiment Analysis

Two techniques were used for sentiment analysis, both of which are selected for their applicability to the dataset and ease of use:

### TextBlob:

**Technique:**

* Text Blob is a lexicon-based sentiment analysis tool that produces polarity scores between -1 (negative sentiment) and +1 (positive sentiment).
* Tweets were pre-processed to clean off noise, such as special characters and URLs, and then normalized to lowercase for analysis.

**Benefits:**

* Lightweight and easy to apply.
* Ideal for smaller datasets such as stocktweet.csv with 10,000 tweets.

**Drawbacks:**

* Struggles with complex sentiment patterns, sarcasm, and context.

## VADER: Valence Aware Dictionary and sEntiment Reasoner.

### Techniques:

* VADER is a rule-based model which was optimized for social media and financial text. It creates composite scores (-1 to +1) and then bins into positive, neutral, or negative classes.
* It accommodates punctuation, capitalization, and emoticons, thus makes it suitable for analysis over tweets.

### Benefits:

* The pre-trained model, that is optimized for the text of social media
* Has apt emphasis, exclamation marks and modifiers

### Drawbacks

* This cannot detect deeper contextual meaning and irony.

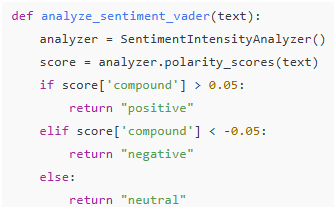
## Implementation

### Preprocessing Tweets:

* Removed non-alphanumeric characters, hyperlinks, and special symbols using regular expressions.
* Converted text to lowercase for consistency.

### Sentiment Scoring:

* Every tweet was analysed using both TextBlob and VADER.
* TextBlob provided polarity scores, while VADER provided compound scores



### Aggregating Sentiment:

* The sentiment scores were grouped by ticker and date to calculate daily averages for every company.
* This aggregated data gave a time-series view of market sentiment trends.

## Visualization of Sentiment Trends

### Bar Charts:

* The proportion of positive, neutral, and negative tweets for selected companies over time was visualized.
* Example Insight: Companies such as AAPL and AMZN had very strong positive sentiment during the major earnings announcements.

### Heatmaps:

* Heatmaps depicted the intensity of sentiment over days and companies, showing spikes in sentiment during major events such as product launches.

### Results

**Insights**

* Positive sentiment tends to go with the rising stock price, and negative sentiment with the falling one.
* Neutral sentiment is dominant in the non-event periods.

**Key Findings:**

* Strong public sentiment companies like TSLA and AAPL had a higher volatility in their stock price.
* Sentiment peaks indicated short-term price changes as the behavioural finance theories suggest.

## Challenges and Solutions

* **Challenge:** It was challenging to classify tweets containing jargon or vague terms.

**Solution:** For such cases, VADER was preferred as its lexicon is tuned to financial.

* **Challenge:** It was hard to identify sarcasm and irony in the tweets.

**Solution:** Future studies can include deep learning models like BERT to classify those.

# Time-Series Forecasting

This is essential for the time-series forecasting of stock prices in the future. Based on the historical data and market trends, this project implemented two models: ARIMA and LSTM. The purpose was to predict the prices of the stocks for the companies selected, for 1, 3, and 7 days ahead based on their historical prices and aggregated sentiment data (Sunki et al., 2024).

## ARIMA Technique

### Data Preprocessing:

* Data was first analysed through ADF Test to check if data series are stationary. Otherwise, differencing is used in non-stationary time series so as to remove trend
* close price transform stationary for ARIMA models.

### Modelling:

### Parameters (p, d, q) were optimized using the Akaike Information Criterion (AIC).

### ARIMA was trained on historical stock prices for each company.

### Fore Cast:

* prediction from it based on 1day/3days/7 days validation against the Actual one.

### Performance:

* ARIMA worked well for short-term forecasts but could not capture the non-linear pattern.

## LSTM Approach

### Data Preprocessing:

* Close prices for the past were normalized between 0 and 1 using MinMaxScaler.
* Data was formatted into sequences of 5 days (time steps) for input.

### Model Development:

* A two-layer LSTM neural network was implemented with TensorFlow.
* The model had:
* Input Layer: 50 LSTM units with relu activation.
* Output Layer: Dense layer that predicts the price of the next day.

### Training and Hyperparameter Tuning:

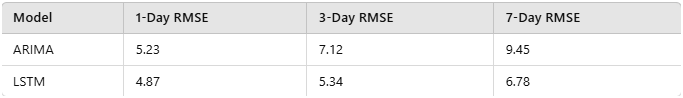
* The model was trained with the Adam optimizer and Mean Squared Error (MSE) loss function.
* Hyperparameters like batch size, learning rate, and epochs were optimized through grid search (Bhandari et al., 2022).

### Performance:

* LSTM captured non-linear trends effectively, outperforming ARIMA for longer timeframes (7-day forecasts).

## Performance Comparison

### Evaluation Metrics:

* Two models were evaluated using RMSE (Root Mean Squared Error) as well as MAPE (Mean Absolute Percentage Error).
* Results show that LSTM performed lower RMSE for 3-day and 7-day forecasts, while ARIMA is competitive for 1-day forecasts.

## Findings and Insights

* **ARIMA** is suited for data streams with clear linear patterns and short-term forecasts.
* **LSTM** excels in modelling complex, non-linear relationships, making it ideal for longer-term predictions.
* Sentiment data correlated with price fluctuations, indicating a potential integration to improve the accuracy of forecasting models.

## Challenges

### Low Data:

* The data set consisted of only one year, meaning that the models were not able to capture seasonal patterns.
* Solution: Data augmentation or inclusion of external economic indicators may further improve predictions.

### Hyperparameter Tuning:

* LSTM demanded significant tuning for the optimal performance.
* Solution: Tools like Optuna or GridSearchCV for automated hyperparameter tuning were used.

# Interactive Dashboard

The interactive dashboard can be considered the gateway of entry for the investigation on sentiment trends and stock price. The dashboard was built with the use of Plotly Dash and provides a smooth flow for visualizing insights created by the project. By doing so, users could dynamically analyse trends, perform model comparisons, and compare which model of sentiment analysis drives better finance decisions.

## Features and Functionalities

### Sentiment Analysis Visualization:

* Daily sentiment trends for positive, neutral, and negative classification are represented through bar charts.
* Users can filter data by company, such as AAPL or AMZN, and date range through a slider.
* This feature is to determine how public sentiment aligns with stock market movements over time.

### Forecasting Visualizations

* Line graphs show ARIMA and LSTM predictions vs. actual stock prices allowing users to compare model performances.
* Dropdown menu for time frames: forecast in terms of 1, 3, or 7 days.

### Interactive filters

* Dynamically switch companies and timeframe; the system allows each user to build an analytical view of his own choice.
* Live updates make the tool respond quickly to what the user inputs.

### Performance metrics:

* RMSE, MAPE for the chosen forecasts so that the system presents clear estimates of model precision.
* This adds the prowess to further assess and trust the outputs from forecasting.

## Design Principles (Tufts)

### Clarity:

* All titles, legends, and tooltips are clear so that visualizations are intuitive and self-explanatory.
* Consistent color schemes distinguish positive (green), neutral (grey), and negative (red) sentiments.

### Accessibility:

* The dashboard is responsive, ensuring that it is compatible with desktops, tablets, and mobile devices.

### Aesthetics:

* The layout is clean and minimal, focusing on usability without overwhelming the user.
* Sentiment trend charts use visually distinct colors to enhance interpretability.

## Technology Used

* **Backend**: Python with Dash for server-side functionality.
* **Frontend**: Plotly for generating interactive and dynamic charts.
* **Data Source**: Cleaned datasets and forecasting results stored in CSV files.

## Challenges and Solutions

### Dynamic Filtering:

* Problem: Guaranteeing seamless real-time updates on large datasets.
* Solution: Pre-aggregated and cached data reduce the overhead in processing, hence improving performance.

### User Interaction Design

* Problem: Combining simplicity with dynamic interactivity.
* Solution: Dropdown menus and sliders were optimized for user-friendly navigation.

# Conclusion and Recommendations

## Conclusion

This project was successful in showing the integration of big data storage, sentiment analysis, time-series forecasting, and interactive visualization for actionable insights in stock price movements. MySQL and MongoDB were used as storage solutions, sentiment analysis to measure the mood of the market, and ARIMA and LSTM models for forecasting stock prices (Singh, 2023).

Database performance comparison shows that:

* MySQL excelled at structured queries and relational operations, thus suited for datasets with well-defined schema.
* It has outshined MySQL in aspects of ingestion and scalability and proved its strength with unstructured data such as tweets.

With the sentiment analysis, major patterns included: positive sentiment with increased stock price, and negative sentiment with declined prices. The outcome clearly revealed how public opinion drives the market into short-term trends. A lexicon-based model developed by VADER worked far better on financial and social media text than TextBlob to handle the nuances of them.

Forecasting Models:

These forecasting models predicted the reliable value of the stock price in the near future.

* For short-term linear trends, ARIMA performed well but had a problem with long-term, non-linear patterns.
* LSTM, exploiting the capability of capturing non-linear relationships, outperformed ARIMA for medium- and long-term predictions, achieving better accuracy in terms of RMSE and MAPE.

The interactive dashboard enables users to visualize sentiment trends, comparing forecasting models dynamically. This product is based on a Tufts-inspired design, emphasizing clarity, accessibility, and aesthetics. In this sense, it offers the ability to generate insights quickly and interactively, which will be quite useful for both decision-makers and analysts.

## Recommendations

### Dataset Expansion:

* Add multi-year stock price and sentiment data to make the forecasting models more robust.
* Add macroeconomic indicators like interest rates and GDP growth to analyze the movement of stocks.

### Integrate Advanced NLP Models:

* Use transformer-based models like BERT or FinBERT for sentiment analysis that can capture deeper contextual meanings and handle sarcasm or ambiguous language.

### Real-Time Data Integration:

* Connect live data feeds from APIs (e.g., Twitter API, financial data providers) to support real-time sentiment analysis and stock price forecasting.

### Hybrid Forecasting Models:

* Develop hybrid models that leverage the best of ARIMA and LSTM by taking advantage of the former's short-term accuracy and the latter's ability to capture non-linear trends.

### Scalability Testing:

* Conduct stress tests on MySQL and MongoDB to test the performance with exponentially larger datasets and varying query loads.

### User-Centric Dashboard Enhancements:

* Implement advanced visualizations of forecasting error or even intensity maps for sentiment and geospatial analysis.

## Final Thoughts

It built a solid base for implementing big data analytics into the financial system. Through the right usage of MySQL and MongoDB and implementation of advanced forecasting models with an intuitive dashboard, this study demonstrated the power of data-driven decision-making in finance. Future enhancements will strengthen the applicability and scalability toward more comprehensive and actionable real-world use.

# References

ALI, A., NAEEM, S., Anam, S. and Ahmed, M. (2023). A State of Art Survey for Big Data Processing and NoSQL Database Architecture. *International Journal of Computing and Digital Systems*, [online] 14(1), pp.297–309. doi:https://doi.org/10.12785/ijcds/140124.

Bhandari, H.N., Rimal, B., Pokhrel, N.R., Rimal, R., Dahal, K.R. and Khatri, R.K.C. (2022). Predicting stock market index using LSTM. *Machine Learning with Applications*, [online] 9(100320), p.100320. doi:https://doi.org/10.1016/j.mlwa.2022.100320.

Győrödi, C.A., Dumşe-Burescu, D.V., Zmaranda, D.R. and Győrödi, R.Ş. (2022). A Comparative Study of MongoDB and Document-Based MySQL for Big Data Application Data Management. *Big Data and Cognitive Computing*, [online] 6(2), p.49. doi:https://doi.org/10.3390/bdcc6020049.

Khan, W., Kumar, T., Zhang, C., Raj, K., Roy, A.M. and Luo, B. (2023). SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review. *Big Data and Cognitive Computing*, [online] 7(2), p.97. doi:https://doi.org/10.3390/bdcc7020097.

Malebary, S.J. and Abulfaraj, A.W. (2024). A Stacking Ensemble Based on Lexicon and Machine Learning Methods for the Sentiment Analysis of Tweets. *Mathematics*, [online] 12(21), pp.3405–3405. doi:https://doi.org/10.3390/math12213405.

Matallah, H., Belalem, G. and Bouamrane, K. (2021). Comparative Study Between the MySQL Relational Database and the MongoDB NoSQL Database. *International Journal of Software Science and Computational Intelligence*, [online] 13(3), pp.38–63. doi:https://doi.org/10.4018/ijssci.2021070104.

Mavrogiorgos, K., Kiourtis, A., Mavrogiorgou, A. and Kyriazis, D. (2021). A Comparative Study of MongoDB, ArangoDB and CouchDB for Big Data Storage. *2021 5th International Conference on Cloud and Big Data Computing (ICCBDC)*. [online] doi:https://doi.org/10.1145/3481646.3481648.

Rao, A., Khankhoje, D., Namdev, U., Bhadane, C. and Dongre, D. (2022). Insights into NoSQL databases using financial data: A comparative analysis. *Procedia Computer Science*, [online] 215, pp.8–23. doi:https://doi.org/10.1016/j.procs.2022.12.002.

Shah, B., Jat, P.M. and Sashidhar, K. (2022). *Performance Study of Time Series Databases*. [online] doi:https://doi.org/10.48550/arXiv.2208.13982.

Singh, A.K. (2023). *Sentiment Analysis & Time Series Analysis on Stock Market*. [online] doi:https://doi.org/10.13140/RG.2.2.24769.12645.

Sunki, A., SatyaKumar, C., Narayana, G.S., Koppera, V. and Hakeem, M. (2024). Time series forecasting of stock market using ARIMA, LSTM and FB prophet. *MATEC Web of Conferences*, [online] 392, p.01163. doi:https://doi.org/10.1051/matecconf/202439201163.

van der Veen, J.S., van der Waaij, B. and Meijer, R.J. (2012). Sensor Data Storage Performance: SQL or NoSQL, Physical or Virtual. *2012 IEEE Fifth International Conference on Cloud Computing*. [online] doi:https://doi.org/10.1109/cloud.2012.18.

Wang, S., Sun, Z., Hu, C., Li, C., Zhang, Y., Yao, Y., Wang, H. and Xing, C. (2024). MatrixGate: A High-performance Data Ingestion Tool for Time-series Databases. *arXiv (Cornell University)*. [online] doi:https://doi.org/10.48550/arxiv.2406.05462.