Chronos Trade: Multimodal Stock exchange report

Bibaswan Biswas

December 2025

Abstract

In this project, we cover ML concepts like ARIMA,LSTM etc to make time series forecast like stock price prediction.

Contents

1 Week-1:ARIMA Model

In the first week, we studied about the ARIMA Model which is a statistical model used to make time series prediction that accounts:

- Autoregression(AR)
- Differencing(I)
- Moving Average(MA)

1.1 Autoregression

Autoregression refers to something like a moving best fit, that is, it is the fitting of data based on previous data points at past time. The model is therefore of the form:

$$x_t = \sum_{j=1}^{p} \phi_j x_{t-j} + w_t$$

where x_i is the observation made at time i. The ϕ_j are the autoregressive parameters and w_t is a noise term for time t. The value p is called the order of the AR.
Thus, AR involves finding the relation among the previous data points to fit the new data point.

1.2 Differencing

Stationarity: For ARIMA, we need to ensure the data does not have too much local variations. Hence, we need the data to be stationary, that is the mean and variance should be close to zero and the covariance between two data points should not depend on the individual time of the observation even though it may depend on the difference between the times.
The need for this is basically that the ARIMA is linear in no of past observations considered and hence it certainly cannot model data with high fluctuations.
Hence, in order to correct the data for stationarity, we use the technique of **differencing**, which is basically taking the difference between consecutive data points, i.e.,

$$x'_t = x_t - x_{t-1}$$

Thus, it is the change in the observed values over time. The number of times we need to use differencing to make the data stationary is denoted by d and is called the order of differencing.

1.3 Moving average

The Moving average(MA) part of ARIMA describes the dependency of the current observation on the previous forcast errors. The model is of the form:

$$x_t = \sum_{j=1}^{q} \phi_j w_{t-j} + w_t$$

where w_i is the noise in the observation at time i. The number of terms in this model, denoted by q is called the order of the MA.

1.4 ACF/PACF

ACF is the autocorrelating function,which is mathematically given by

$$\rho(k) = \frac{Cov(x_t, x_{t-k})}{Var(x_t)}$$

which is basically a measure of how much the time series resembles itself after k time steps. The PACF, is just the partial ACF, which is the ACF due to a lag,say k, with the effects of all shorter lags removed.

1.5 About the code

In the code, I used the weekly closing stock price of Tesla over 4 years for the ARIMA analysis. In the code, f is the fraction used as training data, currently set at 0.8 . At first the data is differenced until it becomes stationary. The number of times differenced gives us the order of differencing, d. Once it is stationary, we find the ACF and PACF plots. After that we find the optimum orders of AR and MA, p and q respectively such that with the pre obtained d, we get minimum AIC(Akaike Information Criterion), i.e., 2(no of paramenters - ln(maximum value of Likelihood function)) . Once, we get the optimal p,q,d we train the model over the training dataset. Finally we forecast the model for the remaining time stamps and calculate the error from actual and forecasted data.

1.6 Analysis and Limitation

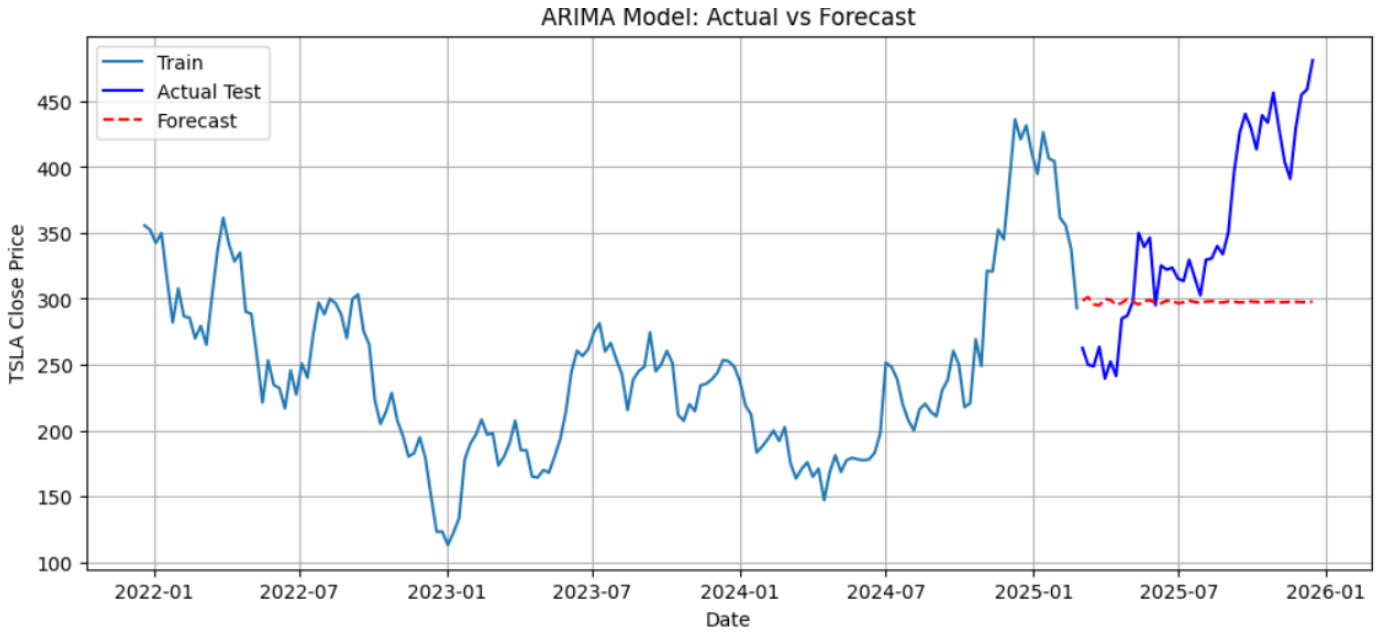


Figure 1: Analysis of Tesla weekly stock closing price

From the final model, we found that the prediction is almost horizontal, however the actual data is very fluctuating. The ARIMA model is not quite accurate as it ignores local variations and needs stationary data. Upon some research, found that LSTM would overcome the shortcomings of ARIMA.

2 Week-2:LSTM Model

2.1 Concept behind LSTM

LSTM (Long Short-Term Memory) is a special type of Recurrent Neural Network (RNN) designed to handle long-term dependencies in sequential data. Traditional RNNs often suffer from the vanishing gradient problem, which makes it difficult for them to learn patterns over long time intervals. LSTM addresses this issue by introducing a memory cell and gating mechanisms.

An LSTM unit consists of three main gates: the input gate, forget gate, and output gate. The input gate controls how much new information from the current time step should be stored in the cell state. The forget gate decides which information from the previous cell state should be retained or discarded. The output gate determines how much information from the cell state is passed to the output. This gating structure allows the LSTM to selectively remember important information and forget irrelevant details, making it well-suited for time series forecasting tasks such as stock price prediction.

2.2 ARIMA and LSTM Models for Time Series Forecasting

ARIMA (AutoRegressive Integrated Moving Average) is a classical statistical approach for time series forecasting. It models future values based on a linear combination of past observations and past forecast errors. ARIMA assumes that the time series is stationary, or can be made stationary through differencing.

LSTM, in contrast, is a deep learning-based model that learns patterns directly from data without requiring explicit assumptions about stationarity or linearity. By using past sequences of data as input, LSTM can capture complex temporal relationships, including non-linear trends and long-range dependencies.

2.3 Differences Between Statistical and Deep Learning Approaches

Statistical models such as ARIMA rely on mathematical assumptions about the data, including stationarity and linear behavior. These models are relatively simple to interpret and require less data for training. However, their performance can be limited when the underlying data exhibits non-linear patterns.

Deep learning models like LSTM are data-driven and capable of modeling complex, non-linear relationships. They generally provide better performance on large and complex datasets but require more computational resources and careful hyperparameter tuning. Additionally, deep learning models are often harder to interpret compared to statistical models.

2.4 Important Concepts

2.4.1 Stationarity and Differencing

A time series is considered stationary if its statistical properties, such as mean and variance, remain constant over time. Stationarity is a key requirement for ARIMA models. Since stock price data is usually non-stationary, differencing is applied to remove trends and stabilize the mean. Differencing involves subtracting the previous observation from the current observation.

2.4.2 ACF and PACF Plots

Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots are used to identify suitable parameters for the ARIMA model. The ACF plot helps in determining the order of the moving

average (MA) component, while the PACF plot assists in selecting the order of the autoregressive (AR) component. These plots provide a visual understanding of the relationship between current and past values at different lag intervals.

2.4.3 Sliding Window Technique for LSTM

LSTM models require input data in a supervised learning format. The sliding window technique is used to convert time series data into sequences, where a fixed number of past observations are used to predict the next value. This approach allows the LSTM to learn temporal dependencies across multiple time steps.

2.4.4 Data Normalization

Data normalization is an important preprocessing step when training neural networks. Large variations in data values can slow down training and affect model convergence. Normalization techniques such as Min-Max scaling are commonly used to scale the data within a fixed range, improving training stability and performance.

2.5 Insights from Implementation

2.5.1 Challenges Faced

Selecting appropriate parameters for the ARIMA model was challenging and required multiple iterations using ACF and PACF plots. Ensuring stationarity through differencing also required careful analysis.

For the LSTM model, determining the optimal window size, number of layers, and training epochs was non-trivial. Overfitting was observed when the model was trained for too many epochs on limited data.

2.5.2 Observations from Model Performance

ARIMA was effective in capturing short-term linear trends but struggled during sudden fluctuations in stock prices. LSTM demonstrated better adaptability to complex patterns and non-linear movements, although it required longer training time.

2.5.3 Comparison of Prediction Accuracy

Overall, LSTM produced more accurate forecasts compared to ARIMA, particularly during volatile periods. ARIMA predictions were more stable and easier to interpret, while LSTM forecasts were more flexible and responsive to changes in the data.

2.6 Visualization and Output Analysis

Stock price versus time plots were used to observe overall trends and volatility in the data. Forecast comparison plots showed the difference in predictive behavior between ARIMA and LSTM models. Residual plots were used to analyze model errors, where ARIMA residuals showed more structure, while LSTM residuals appeared more random. The LSTM learning curves, showing training and validation loss, provided insights into model convergence and potential overfitting.

Figure 2: Using LSTM for stock market prediction

2.7 Conclusion

This work compared ARIMA and LSTM models for time series forecasting. ARIMA performed well for simpler and more stable patterns, while LSTM was better suited for capturing complex and non-linear relationships in stock price data.

ARIMA is computationally efficient and interpretable but limited by its assumptions. LSTM is powerful and flexible but requires more data and careful tuning. Future improvements could include using advanced LSTM architectures such as Bidirectional LSTM or GRU, incorporating additional features like trading volume or technical indicators, or exploring hybrid ARIMA-LSTM models.

3 Week 3–4: Sentiment Analysis and Stock Market Prediction

3.1 Sentiment Analysis

3.1.1 Objective

The objective of this task is to understand and apply sentiment analysis techniques using the `TextBlob` library. Given a dataset containing textual reviews, each review is analyzed to determine its sentiment polarity and classified into one of three categories: positive, neutral, or negative. The analysis focuses on understanding the overall sentiment distribution and identifying general patterns present in the textual data.

3.1.2 Dataset Description

The dataset consists of textual reviews provided in a tabular format. Each entry contains a review written in natural language. The dataset was preprocessed to remove missing values and ensure compatibility with the sentiment analysis pipeline.

3.1.3 Methodology

Sentiment analysis was performed using the `TextBlob` Python library. For each review:

- The sentiment polarity score was computed using TextBlob's built-in sentiment analyzer.
- Based on the polarity score, sentiment labels were assigned as follows:
  - Positive: polarity > 0
  - Neutral: polarity $= 0$
  - Negative: polarity < 0
- A new column named `Sentiment` was added to the dataset containing the assigned labels.

3.1.4 Results and Analysis

After sentiment classification, the distribution of sentiment labels was analyzed. The dataset exhibited a mix of positive, neutral, and negative sentiments, indicating varied user opinions.

The sentiment distribution can be summarized as:

- **Positive**: Majority of the reviews expressed favorable opinions.
- **Neutral**: A moderate number of reviews were informational or opinion-neutral.
- **Negative**: A smaller portion of reviews reflected dissatisfaction or criticism.

The percentage distribution of each sentiment category was computed to provide a quantitative overview of public opinion reflected in the dataset.

3.2 Stock Trend Prediction Using Sentiment and Time Series Analysis

3.2.1 Objective

The objective of this task is to integrate textual sentiment analysis with financial time series modeling to predict stock price movements. By combining historical stock prices with sentiment derived from financial news, the model aims to capture both quantitative and qualitative market signals.

3.2.2 Dataset Preparation

Two datasets were utilized:

- Historical daily stock price data obtained from Yahoo Finance.
- Financial news headlines corresponding to the same trading dates.

The datasets were cleaned and aligned by date to ensure that each trading day was associated with the relevant news sentiment.

3.2.3 Sentiment Annotation

Each news headline was analyzed using the same TextBlob-based sentiment analysis approach described earlier. The sentiment scores were aggregated on a daily basis and merged with the stock price dataset, creating sentiment-aware time series data.

3.2.4 LSTM-Based Time Series Forecasting

An LSTM (Long Short-Term Memory) neural network was trained using:

- Historical stock prices
- Daily aggregated sentiment scores

The model was designed to learn temporal dependencies and the influence of public sentiment on price movements. Its performance was evaluated and compared against a baseline model developed in Week 2 that relied solely on historical price data.

3.2.5 Evaluation and Comparison

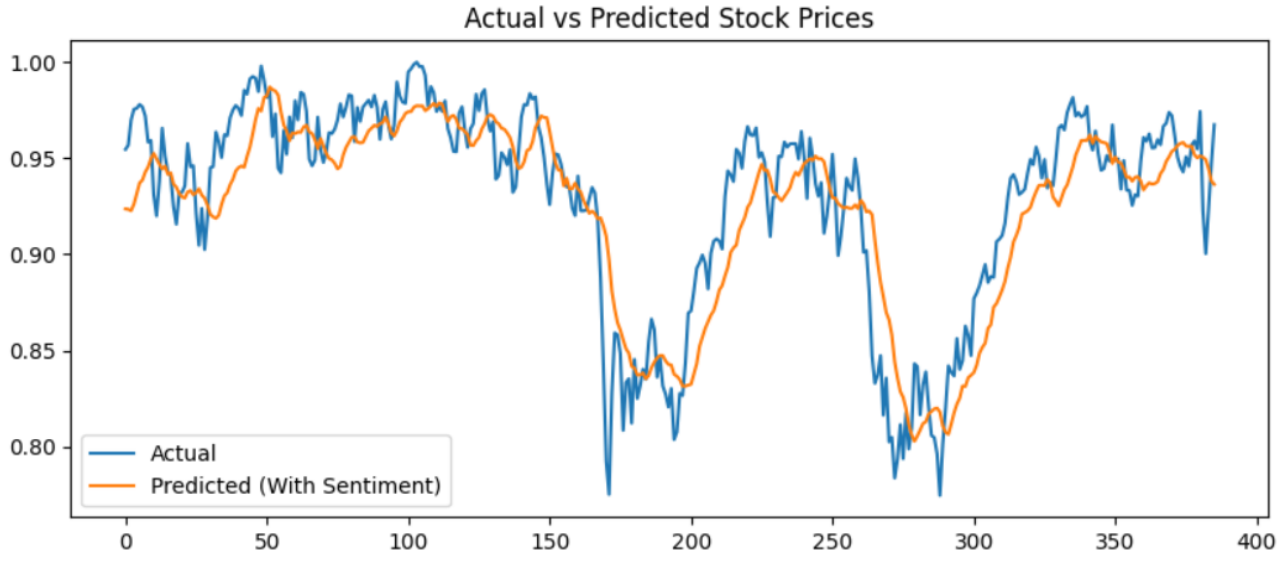
The sentiment-enhanced LSTM model demonstrated improved predictive performance compared to the price-only model. The inclusion of sentiment features helped the model better capture sudden market movements influenced by news events.

3.2.6 Visualization and Interpretation

The following visualizations were generated:

- Comparison of actual versus predicted stock prices
- Trends showing the relationship between sentiment fluctuations and stock price movements

The results indicate that positive sentiment often aligns with upward price movements, while negative sentiment frequently precedes market declines. This highlights the importance of incorporating textual sentiment into financial forecasting models.



3.2.7 Conclusion

This combined approach demonstrates that sentiment analysis, when integrated with time series modeling, can enhance stock price prediction. The experiment confirms that market sentiment derived from news headlines plays a significant role in influencing stock trends and can serve as a valuable feature in predictive financial models.