Applied Forecasting Methods Project Group 6

Raj Shah - 202201403, Ayush Popshetwar - 202201412, Bhoomish Patel - 202201414 and Archi Jariwala - 202201450

Abstract
This report analyzes 1-minute Bitcoin OHLC and volume data (January 2012–present) to model short-term price dynamics. After exploratory data analysis and handling missing intervals, the series is transformed for stationarity and decomposed into trend and seasonal components via STL. ACF/PACF plots guide ARIMA model selection, which is fitted and evaluated using RMSE and MAE on hold-out data. Residual diagnostics confirm model adequacy for intra-day forecasting. Results show ARIMA effectively captures Bitcoin's high-frequency behavior, though extreme volatility suggests opportunities for GARCH or machine-learning enhancements.

1. Problem Statement

Bitcoin, the pioneering cryptocurrency introduced in 2009, has grown into a highly traded digital asset characterized by pronounced volatility and complex market dynamics. Accurate short-term forecasting of Bitcoin's price movements is critically important for traders, portfolio managers, and risk analysts who operate in high-frequency environments. However, the intrinsic non-stationarity of financial time series, the presence of noise at minute-level intervals, and occasional data gaps pose significant modeling challenges. This report addresses the following core problem:

> **How can we effectively model and forecast Bitcoin's one-minute OHLC (Open, High, Low, Close) price series, leveraging exploratory data analysis, time-series transformation, decomposition, and statistical modeling techniques, in order to produce reliable short-term predictions and gain insights into the underlying temporal structures?**

To tackle this problem, we will:

1. **Characterize the data** through exploratory data analysis (EDA) to uncover distributional properties, volatility clusters, and missing-timestamp issues.
2. **Transform and decompose** the raw price series to isolate trend, seasonal, and residual components, assessing stationarity and volatility patterns.
3. **Analyze autocorrelation structures** via ACF and PACF to determine appropriate model orders and understand the memory effects in the series.
4. **Fit and evaluate** various time-series forecasting models (e.g., ARIMA, exponential smoothing), comparing predictive performance using standard error metrics.
5. **Perform residual diagnostics** to validate model assumptions and identify areas for further refinement.

By systematically applying these steps to high-frequency Bitcoin data (sourced from January 2012 to the present), we aim to develop forecasting models that not only achieve competitive accuracy for minute-level price predictions but also enhance our understanding of Bitcoin's dynamic market behavior.

2. Dataset Description

The dataset used in this project is the **Bitcoin 1-minute OHLC (Open, High, Low, Close) time series data**, compiled from various cryptocurrency exchanges. It spans the period from **January 2012 to the present**, providing high-frequency trading data at one-minute intervals.

Each record in the dataset contains the following attributes:

- **Timestamp** – The UTC time corresponding to each 1-minute entry.
- **Open** – The Bitcoin price at the beginning of the minute.
- **High** – The highest price observed within the minute.
- **Low** – The lowest price observed within the minute.
- **Close** – The closing price of Bitcoin at the end of the minute.
- **Volume (BTC)** – The total amount of Bitcoin traded during the minute.

This dataset offers a granular view of Bitcoin's price fluctuations, making it suitable for short-term forecasting and time series modeling tasks. It enables the application of various statistical techniques including transformation for stationarity, decomposition into trend and seasonal components, and model fitting using ARIMA or other approaches.

It is important to note that missing timestamps may exist due to exchange API downtimes or gaps in data collection.

The dataset was originally sourced and published on GitHub via automated scripts that scrape public APIs such as the Bitstamp API. It forms a strong foundation for modeling Bitcoin's intra-day behavior and understanding its volatile nature in a data-driven manner.

3. Exploratory Data Analysis

The data utilized here is the minute-level data for Bitcoin after the year 2012 and consists of 6,990,923 rows with 7 columns. Visually inspecting, we realized that about 0.43% of data contained missing entries (NaNs). We investigated deeper and noticed that the majority of missing values had occurred only in the newer months of 2025, revealing that the data hadn't been updated as fully as required. Since the missing data percentage was low and was limited to the end of the dataset, we decided to remove these rows. This bar chart points out that missing values occurred only in the last rows of the date-time

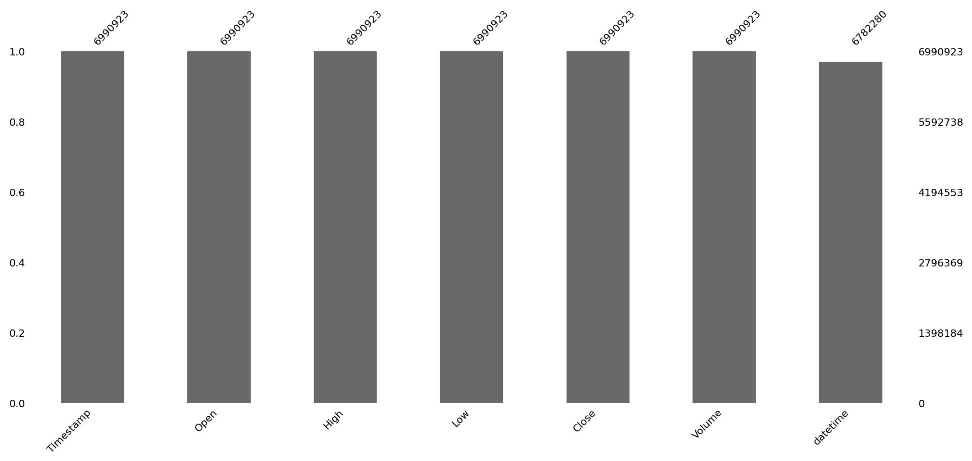


Figure 1: Missing Value Bar Plot

column.

3.0.1. Transition to Hourly Data

To minimize noise and enhance computational efficiency, we resampled the data to hourly levels by averaging the numerical columns. Visual check validated that the general structure and patterns of the time series were retained after resampling. This conversion minimized the dataset to 115,719 rows while keeping all 7 columns.

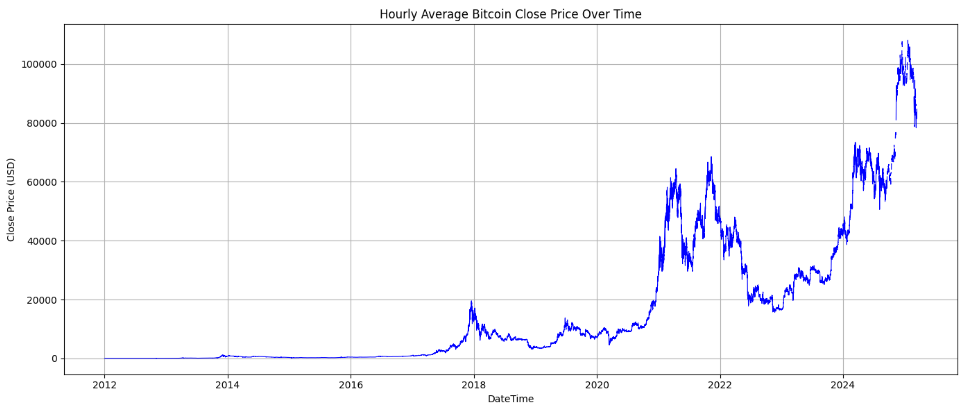
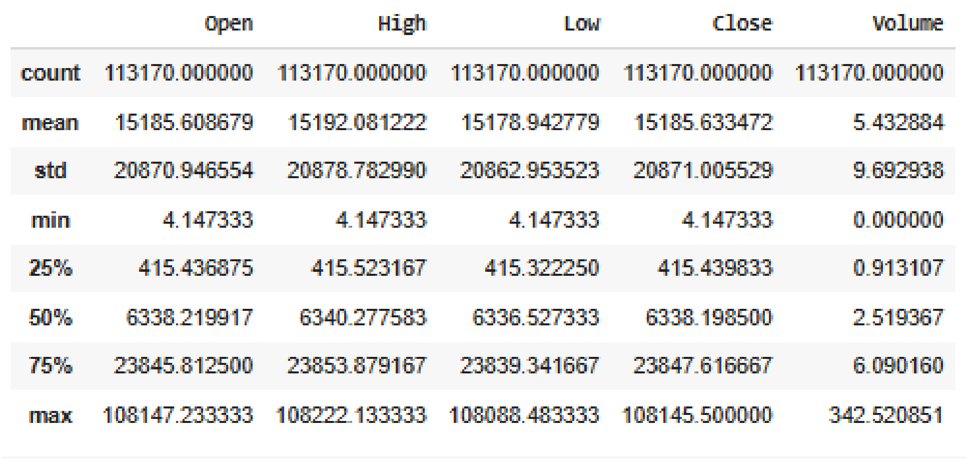


Figure 2: Hourly Time Series Plot

The plot displays the smoothness and continuity of the price trend after resampling.



	Open	High	Low	Close	Volume
count	113170.000000	113170.000000	113170.000000	113170.000000	113170.000000
mean	15185.608679	15192.081222	15178.942779	15185.633472	5.432884
std	20870.946554	20878.782990	20862.953523	20871.005529	9.692938
min	4.147333	4.147333	4.147333	4.147333	0.000000
25%	415.436875	415.523167	415.322250	415.439833	0.913107
50%	6338.219917	6340.277583	6336.527333	6338.198500	2.519367
75%	23845.812500	23853.879167	23839.341667	23847.616667	6.090160
max	108147.233333	108222.133333	108088.483333	108145.500000	342.520851

Figure 3: dfHourly.describe

This table gives important statistics (mean, std, min, etc.) for every feature of the hourly dataset.

3.1. Correlation Analysis

We then calculated the correlation matrix for all features to gauge interdependencies.

Open, High, Low, and Close prices have perfect correlation (correlation coefficient = 1.00), which shows that they move together and practically record the same price behavior. Volume has a weak negative correlation (−0.20) with price-based features, indicating only a weak inverse relationship. The price$_upvariable(likelyabinaryflagforpriceincrease)hasnostronglinearrelationshipwithanyotherfeature,ind$

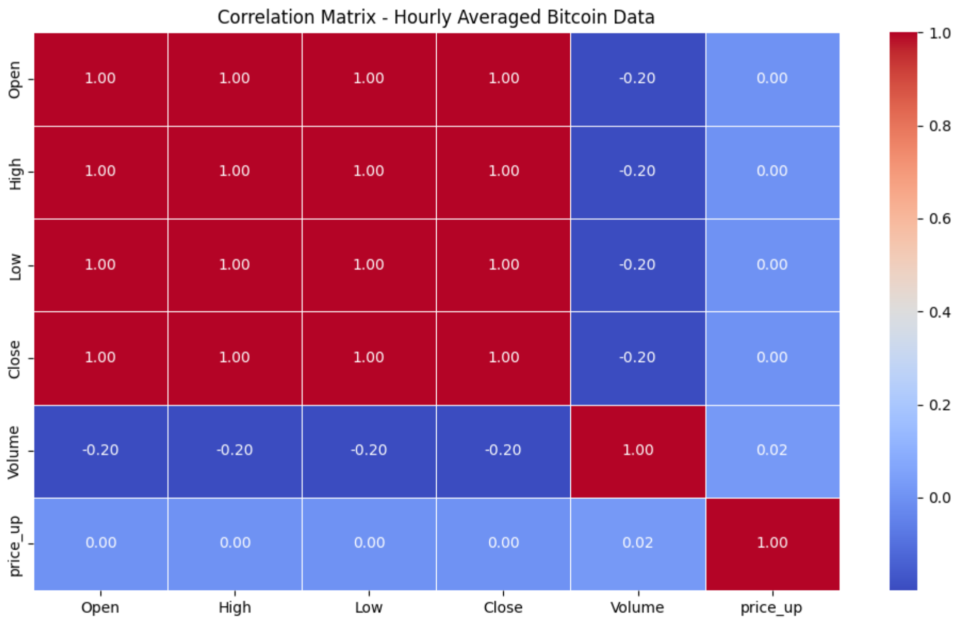


Figure 4: Correlation Matrix

3.1.1. Rationale for Using Close Price as Target

We chose the Close price as our main prediction target because it is a conventional standard in financial analysis, being the most stable and informative value for every period of time. Since there is a perfect correlation between Open, High, Low, and Close prices, more than one of them would only bring multicollinearity and not predictive value. Using the Close price only simplifies models without losing effective representation of market trends.

3.1.2. Why Price Features Are So Similar

In hour-averaged Bitcoin price data, Open, High, Low, and Close prices will be quite close because of:

- Constant and frequent trading on the Bitcoin market.
- Restricted volatility on short time scales (hours), and thus a minimal difference between these prices.
- This is what accounts for the high correlation and redundancy of incorporating all prices into a predictive model.

3.2. Decomposition and Transformation of Time Series

We used the Augmented Dickey-Fuller (ADF) test on the Close price series to determine stationarity. The first ADF test gave a p-value of 0.9807, which is much higher than the conventional value (0.05). This is an indication that the time series was non-stationary and had a unit root.

To counter this, we first-order differenced the Close price. Following differencing, the ADF test statistic fell to -39.54, with a p-value of effectively 0.0. This proves that the differenced series is now stationary since the null hypothesis of non-stationarity is rejected at a very high confidence level.

This plot illustrates how differencing makes the mean of the series stable so that it is ready for time series modeling.

(Fig 6a) The original Close price plot has evident trends and non-stationarity. (Fig 6b) The differenced series after transformation has a stable mean, with consistent statistical properties across time.

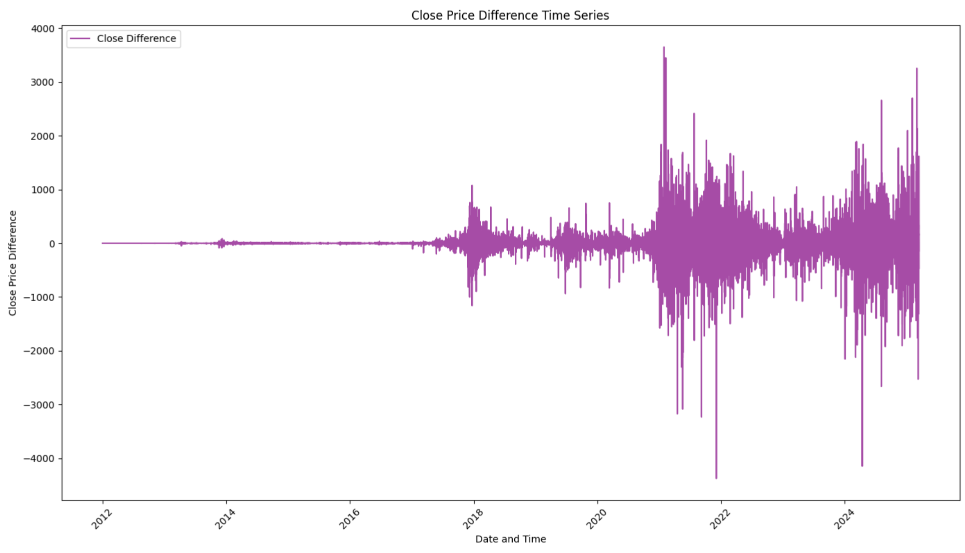
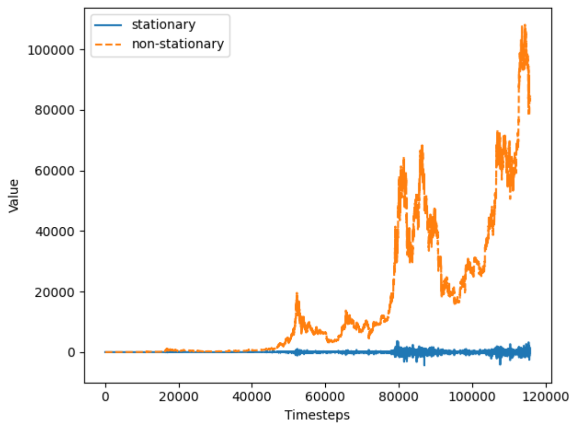
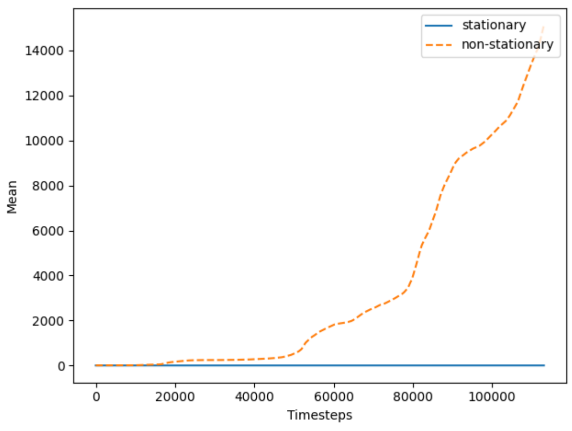


Figure 5: Differenced Close Price Plot



(a) Non-stationary Close price series



(b) Stationary series after differencing

Figure 6: Comparison of original (non-stationary) and transformed (stationary) series. The original Close price (left) shows evident trends and non-stationarity, while the differenced series (right) displays a stable mean and consistent statistical properties across time.

3.3. Seasonal Decomposition

Then, we carried out seasonal decomposition to separate the trend, seasonal, and residual components of the hourly Close price. We employed a seasonal period of 24, based on daily cycles in the hourly data.

- **Trend Component :** There is a strong long-term uptrend evident, reflecting Bitcoin's overall appreciation over time. This trend upwards is consistent with wider adoption, growing investor interest, and its use as a store of value.
- **Seasonal Component:** The seasonal component fluctuates near zero across the series, suggesting that there are no persistent and repeatable patterns in the seasons. Contrary to typical assets (which might display monthly or annual seasonality), the behavior of Bitcoin doesn't seem to conform to regular temporal patterns.
- **Residual Component:**The residual (noise) part exhibits high and jagged spikes, reflecting sudden price movements. They capture Bitcoin's natural volatility and sensitivity to exogenous shocks

like regulatory updates, macroeconomic fluctuations, social media sentiment, and speculative trading activity.

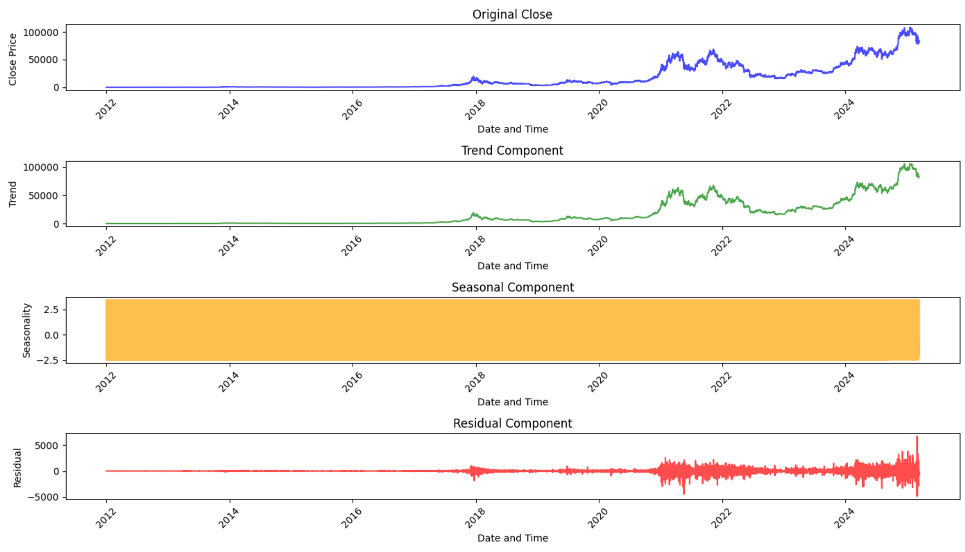


Figure 7: Seasonal Decomposition

Lack of strong seasonality makes time series modeling easier since seasonal adjustments are not required. The very volatile residual component highlights the need to include external sentiment or features for short-term predictions. The trend component gives a smoothed view of long-run growth but can miss swift market movements alone.

3.4. Moving Average Analysis

To see the underlying trend more distinctly, we graphed moving averages with window lengths of 500 and 1000:

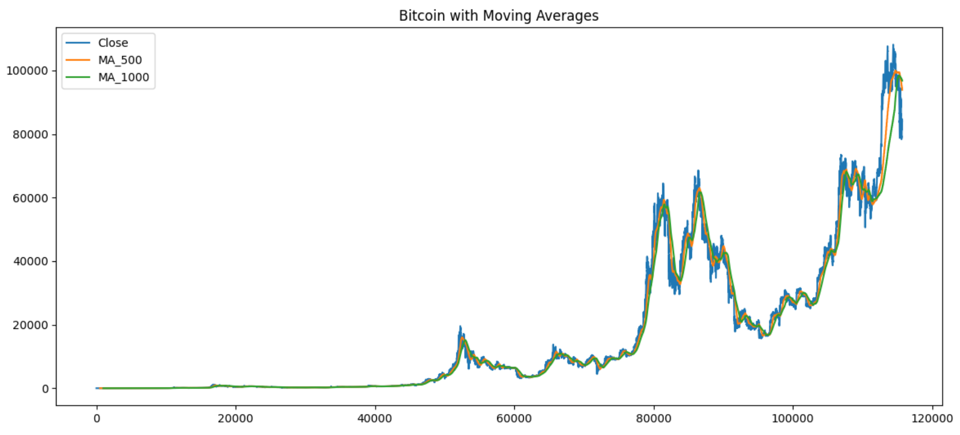


Figure 8: Moving Average (500 1000) Plot

This plot smoothes out short-term oscillations and emphasizes the overall trend in Bitcoin's price. The longer the window, the stronger the smoothing effect, which makes long-term trends easier to see while eliminating noise.

In conclusion, the decomposition and transformation of the Bitcoin time series indicate a definite long-term trend, no persistent seasonality patterns, and extreme volatility propelled by uncertain external influences. This forms the basis for choosing the right modeling methods that incorporate stationarity and volatility but reject seasonality.

3.5. ACF and PACF plots of the series

The Auto-correlation Function (ACF) and Partial Auto-correlation Function (PACF) of a time series help in identifying the orders of a Moving Average (MA) and Auto-Regressive (AR) process the original time series would most likely be following. So, we have plotted the ACF and PACF plots for the feature (Close price diff) to identify the orders of the AR and MA models that would be best to fit a model

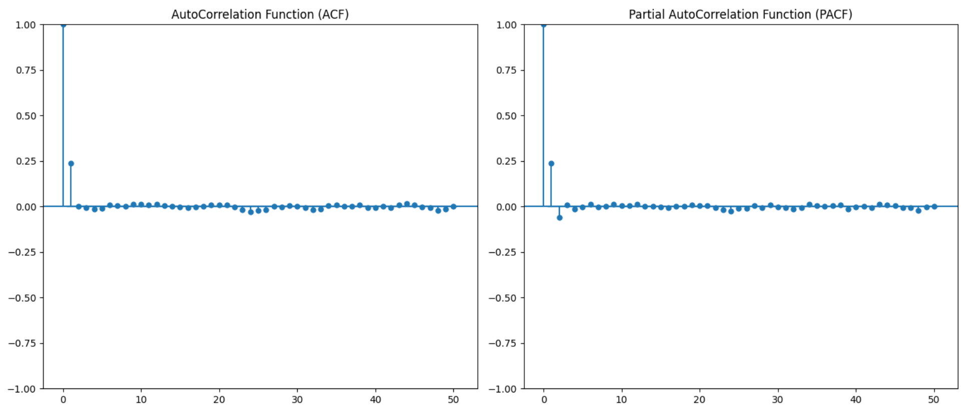


Figure 9: ACF PACF Plot

The ACF plot has a strong spike at lag 2, with later lags falling quickly into statistical irrelevance. This suggests that the current value is strongly correlated with the value two periods in the past, but not very much with values more than two periods in the past. Such a pattern implies the existence of a Moving Average (MA) component, in this case, MA(2), since the ACF captures the correlation of the current value and previous error terms. An abrupt cutoff of the ACF following lag 2 generally implies an MA(q) process with q being the lag at which the cutoff appears.
The PACF graph, however, has a clear spike at lag 2 and an equally steep fall thereafter. This indicates an Auto-Regressive (AR) process of order 2, or AR(2). Unlike the ACF, the PACF separates the immediate correlation between a point in time and its lags after eliminating the impact of intermediate lags. A cutoff in the PACF indicates that after adjusting for the effect of the nearest past values, further lags contribute negligible predictability, typical of an AR process.
Combined, these plots indicate that the differenced close price series is best represented by an ARIMA(2,1,2) model. The differencing part (d = 1) takes care of the non-stationarity of the original series, and the AR(2) and MA(2) terms take care of the short-run memory in the data. If the series were already stationary, the model would be an ARMA(2,2) setup.

4. Model Fitting And Evaluation

For all the models, we split the test data into the last two to four months of the final year for predictions. The training dataset has 112,838 rows, and the test dataset contains 2,881 rows.

4.1. LINEAR MODELS

4.1.1. Moving Average Model

We chose 2 as the order q for the MA model, based on the PACF plot, which showed a significant drop after lag 2, with the values staying within the confidence interval after that. The MA model works when the data depends on past errors, which can be seen through the PACF of the series.

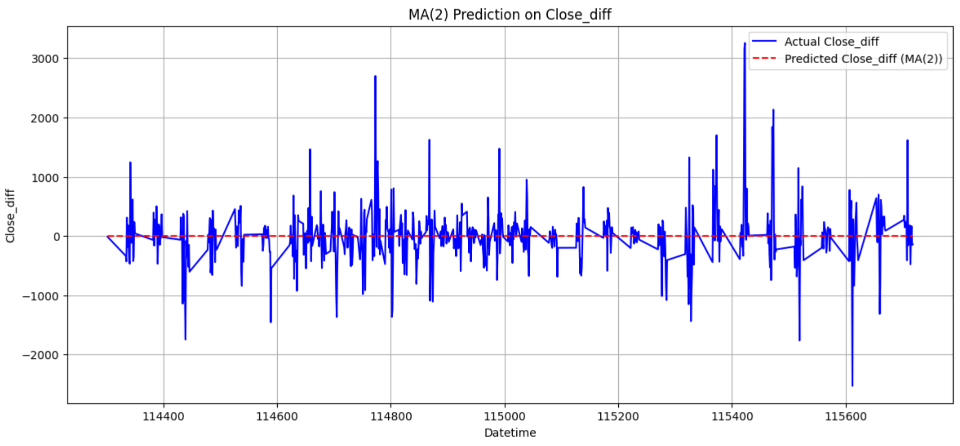


Figure 10: MA Predictions

The plot shows that an MA(2) model doesn't work well for predicting day-to-day changes in Bitcoin's closing price. It gives a poor fit and fails to predict the direction or size of price movements. This could be because Bitcoin's price is hard to predict or the past errors are not enough to explain the price changes. To improve forecasts, more complex models, like ARMA or ARIMA, might be needed.

4.1.2. Auto Regressive Model

For the AR model, we chose order p=2p = 2p=2, based on the ACF plot, which showed a sharp drop after lag 2, with values falling within the confidence interval after that. The AR model is used when the data depends on its past values, which can be seen using the ACF of the series.

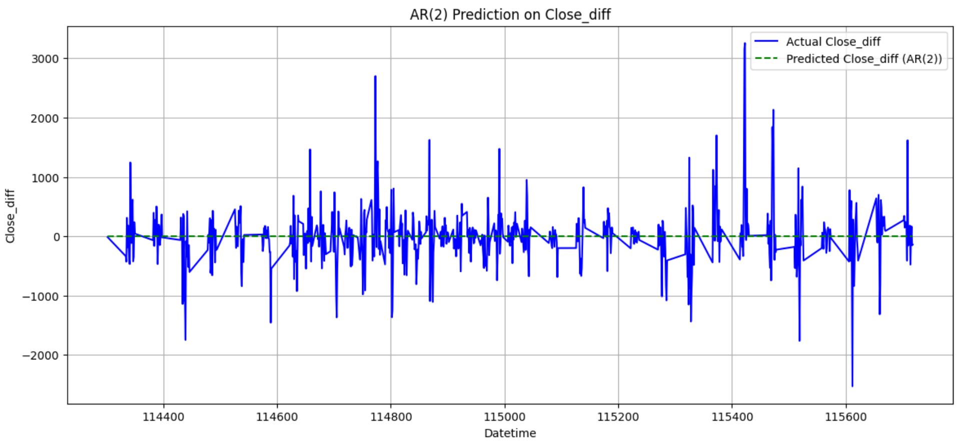


Figure 11: AR Predictions

The AR(2) model gives a small improvement over the flat predictions of the MA(2) model but still doesn't capture the size and timing of the actual price movements well. This suggests that the relationships in Bitcoin's past prices, at least up to lag 2, are not strong enough to make accurate predictions. We may need to look at models with higher AR orders or a combination of AR and MA, such as ARMA or ARIMA.

4.1.3. Auto Regressive Moving Average Model

We chose the ARMA(2,2) model for the same reason as above.

The ARMA(2, 2) model gives a slightly better fit but still doesn't predict Bitcoin's day-to-day price changes well. While it is sensitive to past price movements and errors, it fails to predict the size of

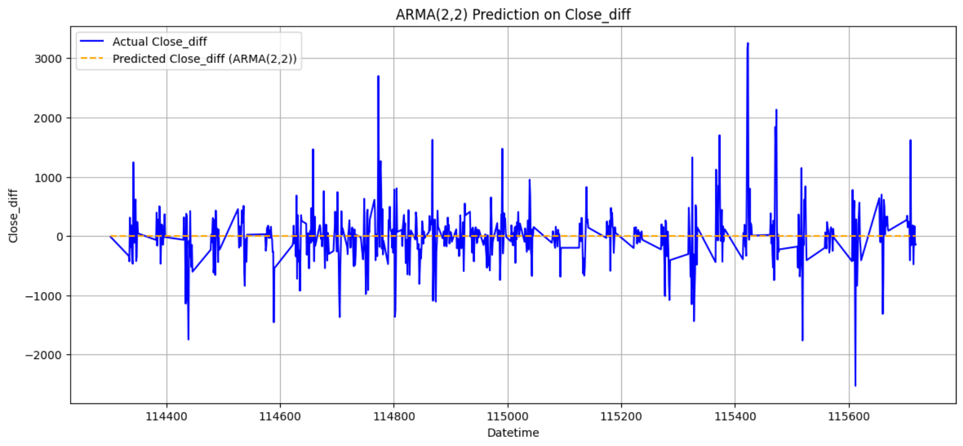
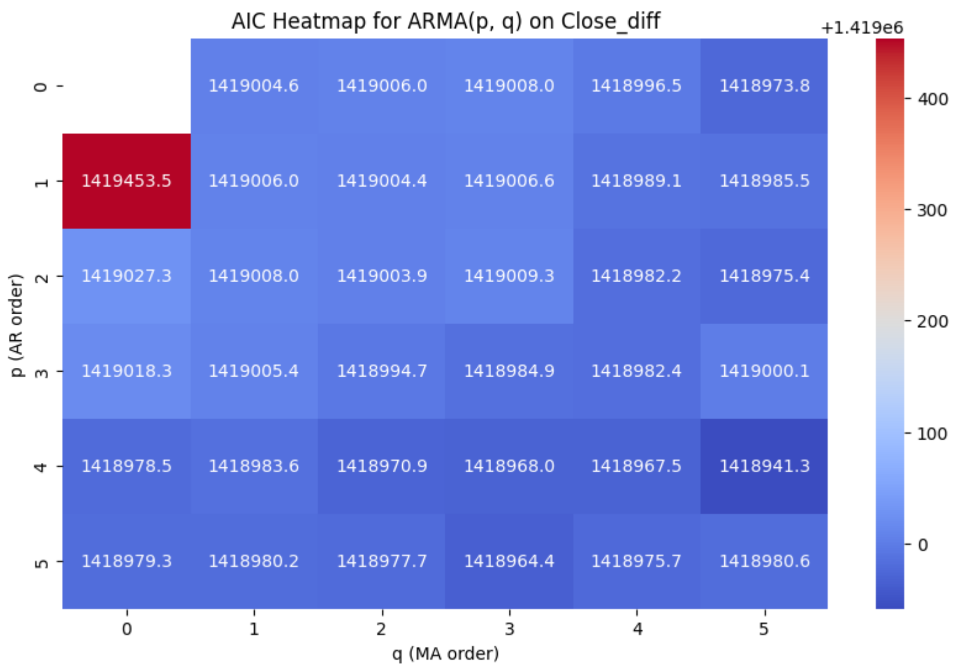


Figure 12: ARMA Predictions

volatility. This suggests that we might need a higher-order ARMA model, models that consider external factors, or non-linear models to make better forecasts for Bitcoin's price changes. The unpredictable nature of Bitcoin and the influence of external events make accurate short-term predictions difficult. We then thought that the model order might not be the best, so we tuned the parameters using the Akaike Information Criterion (AIC), which helps avoid overfitting by balancing model accuracy and complexity. The lower the AIC, the better the model. After tuning, the best order turned out to be (4,5), but it still didn't show much improvement.



p \ q	0	1	2	3	4	5
0		1419004.6	1419006.0	1419008.0	1418996.5	1418973.8
1	1419453.5	1419006.0	1419004.4	1419006.6	1418989.1	1418985.5
2	1419027.3	1419008.0	1419003.9	1419009.3	1418982.2	1418975.4
3	1419018.3	1419005.4	1418994.7	1418984.9	1418982.4	1419000.1
4	1418978.5	1418983.6	1418970.9	1418968.0	1418967.5	1418941.3
5	1418979.3	1418980.2	1418977.7	1418964.4	1418975.7	1418980.6

Figure 13: AIC ARMA Plot

We also analyzed the residuals (the difference between the predicted and actual values) to check if they were normal and uncorrelated, as this is important for fitting a linear model.

- **Significant Autocorrelation Remains**: The ACF and PACF plots of the residuals show a big spike at lag 1, meaning the errors made by the ARMA(2, 2) model are still related to the errors from the previous day. This suggests that the model hasn't fully captured the short-term dependencies in Bitcoin's price changes. Ideally, the residuals should be random and uncorrelated (called white

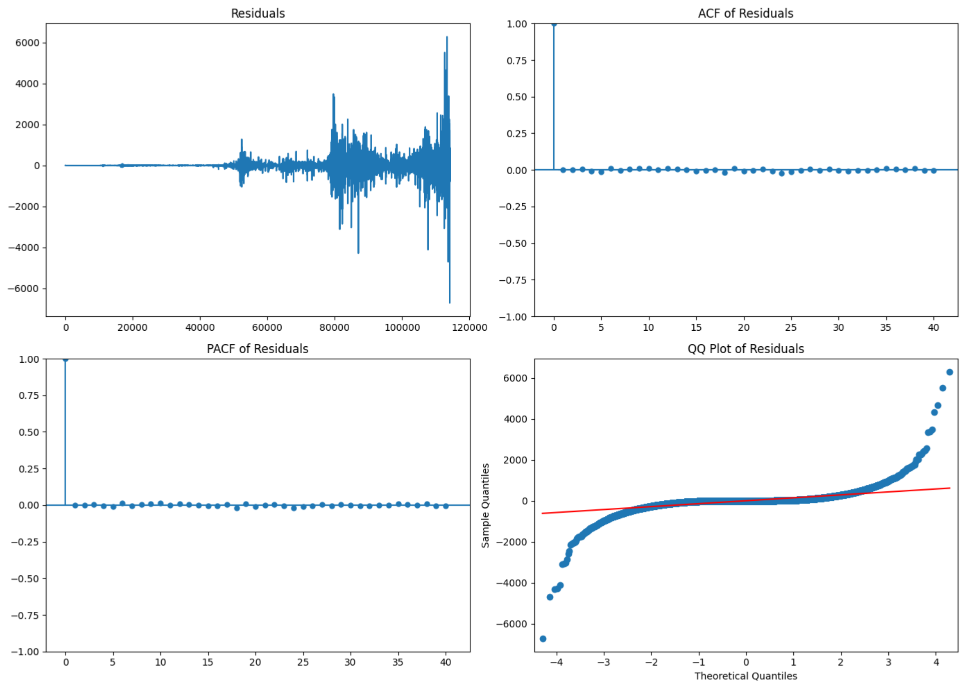


Figure 14: ARMA Residual Plots

noise).

- **Model Isn't Capturing All the Linear Dependencies:**The significant correlation in the residuals shows that there's still useful information in the price changes that the ARMA(2, 2) model hasn't captured. This suggests that the model may be incorrectly specified, and a higher-order ARMA model or a different model might be needed to improve the results.
- **Violation of Model Assumptions:** The non-zero autocorrelations in the residuals go against a key assumption of ARMA models, which is that the errors should be independent and identical (i.i.d.). This could lead to inaccurate estimates and unreliable forecasts.
- **Non-Normal Errors:** The QQ plot shows that the residuals don't follow a normal distribution and have heavier tails than expected. While ARMA models don't need normally distributed errors for estimation, normality is often assumed for statistical analysis, like confidence intervals. The non-normality suggests that extreme price changes happen more often than what a normal distribution would expect.
- **Heteroscedasticity (Non-constant variance):**While this is not clearly visible in the ACF, PACF, or QQ plots, the time series plot of the residuals strongly suggests that the residuals' size changes over time. They become larger during periods of high price volatility, indicating that the model's errors are bigger when the price is more unstable. This breaks the assumption of constant variance.

The residual plots from the ARMA(2, 2) model show that the model is not adequate. The significant correlation in the residuals shows that the model hasn't captured all the linear dependencies in the data. The non-normality and non-constant variance also suggest that the model's forecasts may not be reliable. This highlights the need for a more advanced model to better account for Bitcoin's volatile price changes.

4.1.4. Auto-Regressive Integrated Moving Average (ARIMA)

The main difference between ARIMA and ARMA is that ARIMA includes a differencing step to make the series stationary. So, we performed predictions and fit the model to the differenced closing prices. We used the original closing price after performing one differencing step to make the series stationary. For ARIMA, we again used AIC to determine the best order (4,1,2), with d=1d = 1d=1, meaning we performed one differencing step to make the series stationary. Even with the best model, the results were similar to the ARMA model, and the residual plots showed the same patterns.

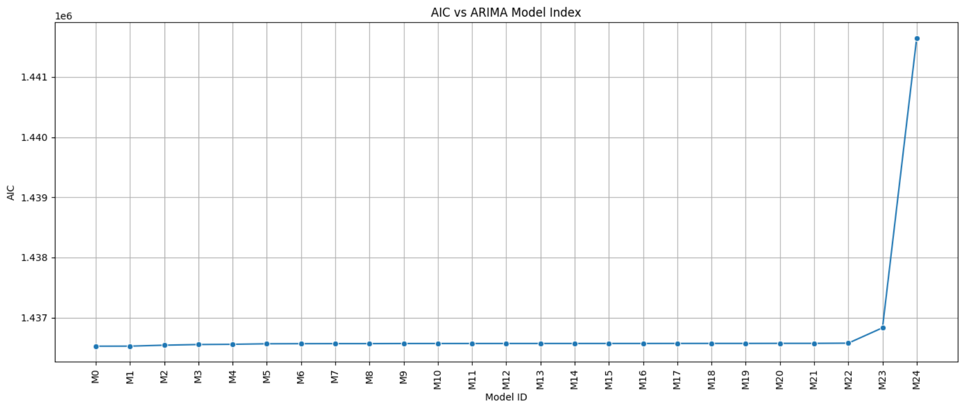


Figure 15: AIC vs ARIMA

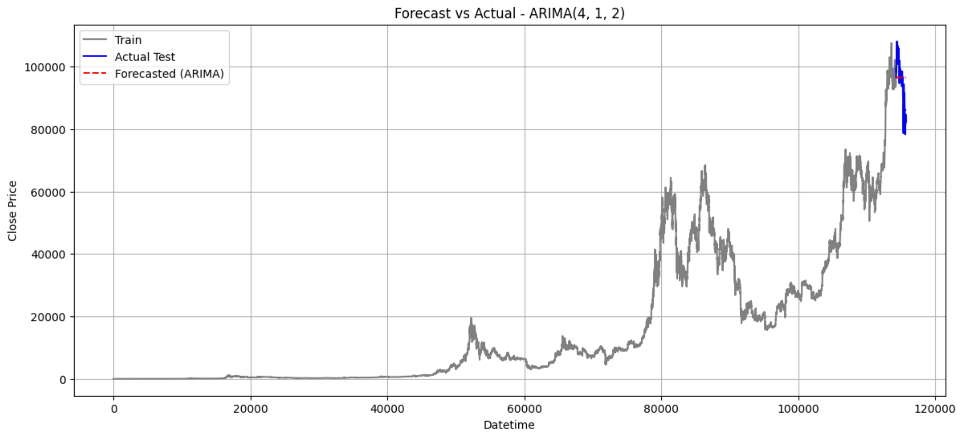


Figure 16: ARIMA Predictions

4.2. Seasonal Auto Regressive Integrated Moving Average (SARIMA)

The SARIMA model is useful when there is seasonality in the time series. However, since no clear seasonality is present in our dataset, we did not expect the model to perform well. As with other models, we used the Akaike Information Criterion (AIC) to determine the best order. The optimal SARIMA order turned out to be:

- Best SARIMA Order: (0, 1, 2)
- Seasonal: (0, 1, 1, 24)

The SARIMA(0, 1, 2)x(0, 1, 1, 24) model, with a seasonal period of 24 (possibly representing hourly data if the overall timeframe is daily), performs poorly in forecasting the Bitcoin closing price on the test

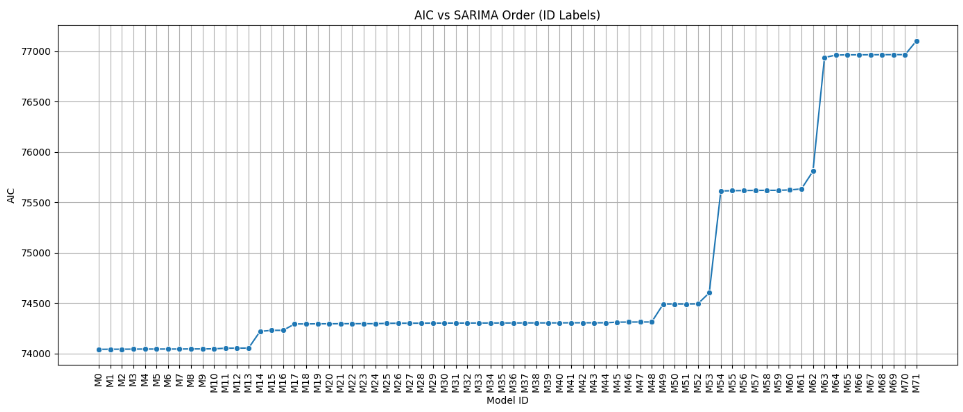


Figure 17: AIC VS SARIMA Plot

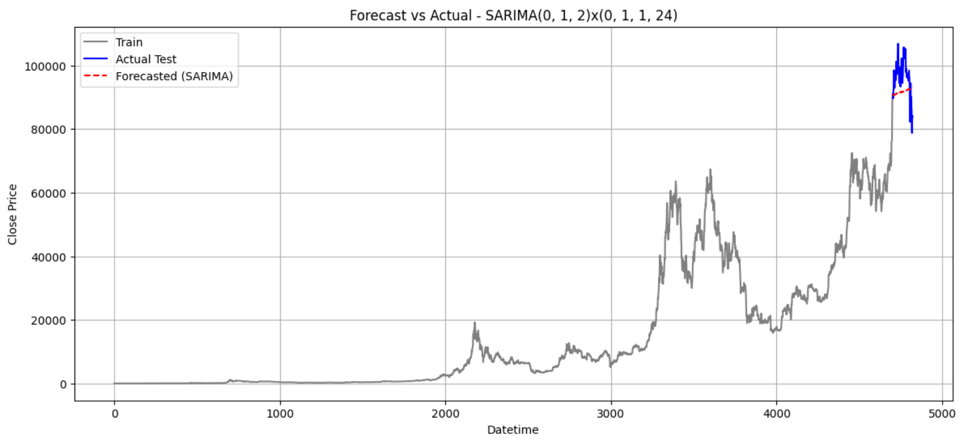


Figure 18: SARIMA Predictions

set. While a visual inspection of the predictions might show some alignment with the final downward trend, the evaluation metrics, especially the highly negative R^2 value, clearly indicate that the model's predictions are unreliable and worse than a naive baseline. The MAE and RMSE provide the magnitude of errors but are overshadowed by the negative R^2, which confirms that the model is performing poorly. The MAPE of 6.81% might seem moderate on its own, but when considered alongside the negative R^2, it becomes clear that this error is occurring in a model that fails to capture the core patterns in the data. The SARIMA model with these parameters is not suitable for forecasting Bitcoin's closing price during this test period. It fails to capture the underlying dynamics, leading to significant errors and performance worse than simply predicting the average price. Further exploration into different model orders, the inclusion of exogenous variables, or alternative forecasting methods would be necessary. The residual plots also suggested similar issues as observed with the ARMA model.

4.2.1. SARIMAX Model

Extending SARIMA, SARIMAX incorporates external variables (exogenous variables) that might influence the target variable. This allows the model to account for additional information beyond the historical values of the target series, potentially improving forecast accuracy. In our implementation of SARIMAX, we have:

- Included Exogenous Variables: Features such as 'pricerange', 'closeopendiff', 'Volume', and 'dayofweek' were used as exogenous variables. These additional predictors allow the model to utilize more data, potentially improving its forecasting ability.

- Implemented Rolling Forecasting: The model is updated iteratively with new data, allowing it to adapt to recent changes in the dataset, which may improve forecast accuracy over time.

We used the Akaike Information Criterion (AIC) again to determine the best model, which turned out to be: Best SARIMAX Order: (2, 1, 1) x (0, 1, 1, 7)

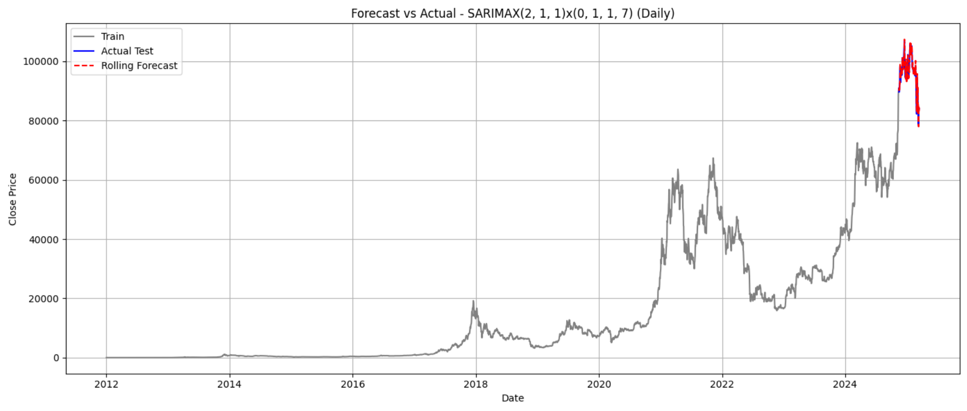


Figure 19: SARIMAX Predictions

The rolling forecast (red dashed line) appears to follow the general direction of the actual test data (blue line) in the short-term. It reacts to some of the upward and downward movements. However, there are noticeable deviations. The forecast doesn't perfectly align with the magnitude of the actual price swings, especially during periods of high volatility. There's still some lag and smoothing in the forecast compared to the sharp changes in the actual price. Inference from Final Evaluation Metrics:

- RMSE (2458.46): The root mean squared error shows a moderate level of prediction error. On average, the model's predictions are off by approximately $2458.46.
- MAE (1640.35): The mean absolute error indicates that the absolute difference between predicted and actual prices is around $1640.35. The MAE is lower than the RMSE, which suggests that large errors are less frequent or less extreme.
- R^2 (0.8218): The R-squared value of 0.8218 is relatively high, meaning that the SARIMAX model explains 82.18% of the variance in the Bitcoin close price during the test period. This represents a significant improvement compared to the previous ARMA models.

The cross-validation results show higher RMSE (12691.80), MAE (9946.37), and a highly negative R^2 (-3.7977). This discrepancy between the cross-validation and final evaluation metrics suggests that the model's performance is highly dependent on the specific train-test split, or that the model might have overfitted to the training data used for the final evaluation.
The correlations of the 'Close' price attribute are of interest. 'Pricerange' has a very strong positive correlation of 0.73, which means that greater daily price ranges are linked to higher closing prices. In contrast, 'closeopendiff' has a weak positive correlation of 0.15, which means that there is a weak correlation between the difference between the close and open prices and the closing price itself. 'Volume' has a moderate negative correlation of -0.31, which means that greater trading volumes are linked to slightly lower closing prices, although this correlation may be more complex and is worthy of further investigation. Finally, 'dayofweek' has a very weak negative correlation of -0.0013, which means that there is almost no linear relationship between the day of the week and the closing price.

The SARIMAX(2, 1, 1)x(0, 1, 1, 7) model with external variables shows a significant improvement in the prediction of the closing price of Bitcoin compared to basic ARMA models. The high R^2 value achieved in the final test reflects a good fit to the test data, explaining much of the variation in the price. Nevertheless, the rolling forecast plot still shows some anomalies, implying that the model is not accurately capturing all the short-term fluctuations.

The significant disparity between the cross-validation and test-set metrics raises some uncertainty regarding the model's stability and vulnerability to overfitting. The cross-validation results demonstrate much worse out-of-sample performance on other sections of the training data. The relations between the features are revealing of the relations between the exogenous variables and the closing price, which the SARIMAX model uses for its forecasting. It appears that 'pricerange' is the most important exogenous feature, as suggested by its correlation.

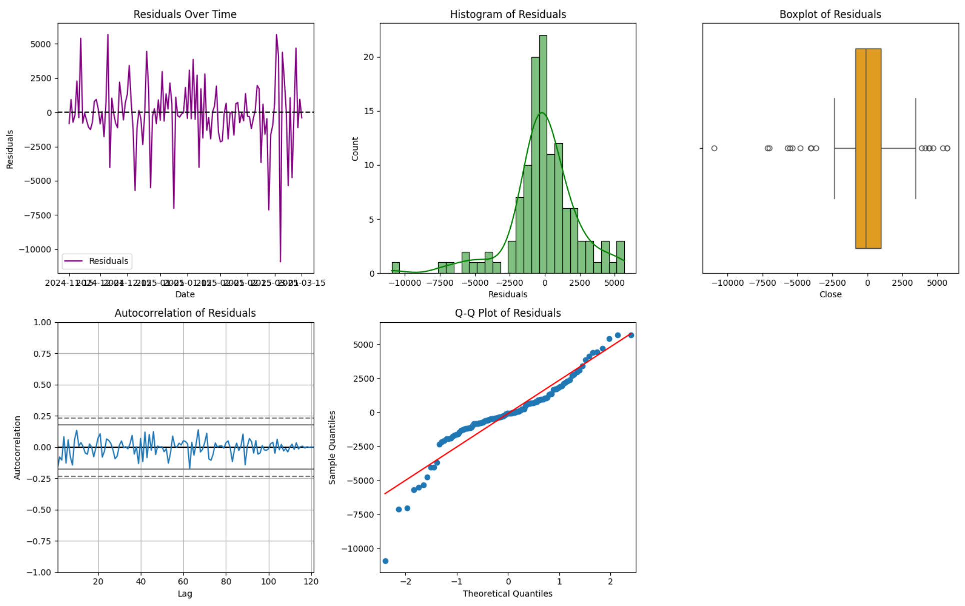


Figure 20: SARIMAX Residuals

The residual plots of the SARIMAX(2, 1, 1)x(0, 1, 1, 7) model provide some interesting insights. The residuals oscillate around zero, i.e., the model is not consistently under- or over-estimating anything. The variability of the residuals is not constant, however, over time, with the errors progressively becoming larger towards the later part of the period, i.e., higher volatility. The large positive and negative spikes are the points where the model has deviated considerably from the realized prices.
The residual histogram is roughly centered around zero but is not symmetrically distributed, and it has some skew and heavy tails. The boxplot similarly has a wide range of residual values and some outliers and indicates large forecast errors. Statistically significant autocorrelation at lag 1 indicates that the model has not adequately accounted for all temporal dependencies. The small spikes at other lags similarly indicate weak autocorrelation.
Q-Q plot also verifies non-normality of residuals since points are not on the red line, indicating heavy-tailed and skewed residuals. Generally, even if the model can perform some pattern fitting, problems such as autocorrelation, non-constant variance, and non-normality in the residuals indicate that it needs to be improved.

4.3. DEEP LEARNING MODELS

4.3.1. RNN

Recurrent Neural Networks (RNNs) are a type of neural network that is particularly suited for sequential data processing. Unlike the standard feedforward neural networks, RNNs have loops in their structure so that they can retain information through a series of time steps. This feature makes them applicable where sequence and context are both significant, e.g., time series forecasting, speech, and natural language processing. In this framework, the RNN is trained on 24 consecutive hourly closing price series (one day) and forecasts the closing price of the subsequent hour. This configuration enables the model to capture temporal patterns and nonlinear relationships within the data without the need for hand-engineered features.

4.3.2. Why RNN

- Temporal Dependency: The financial information such as Bitcoin price depends on the past values. RNNs have been implemented to learn such time-dependent relationships
- Capturing Nonlinear Patterns: Price action is nonlinear and complex. RNNs are able to model these relationships more effectively than simple moving average or linear regression models.
- Sequential Data Processing: RNNs process data sequentially, which is crucial for time series.
- Automatic Feature Learning: They learn meaningful features from sequences without manual feature engineering (e.g., lag variables).

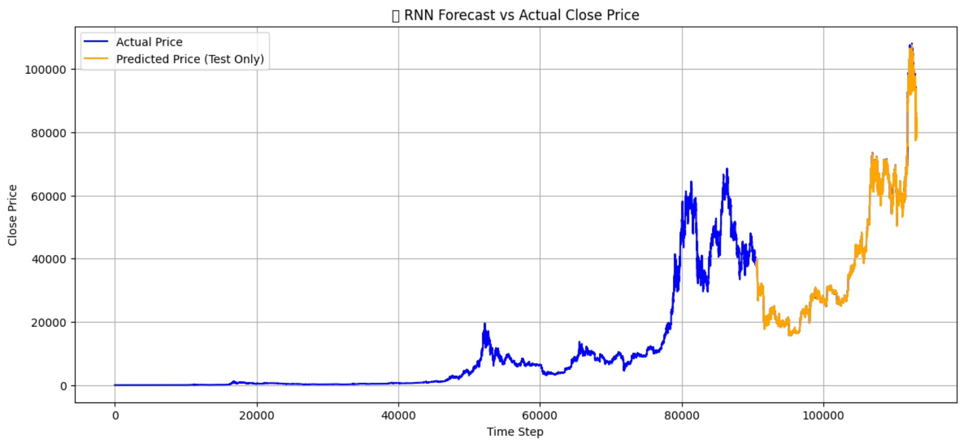


Figure 21: RNN Forecast

4.3.3. Evaluation Metrics

- RMSE (Root Mean Squared Error): 323.2787: It is a measure of the average size of the prediction errors. The smaller the RMSE is, the better the fit.
- R^2 Score: 0.9998: This is almost 1, which means that the RNN model explains about 99.98% of the variation in the Bitcoin closing price over the test period. This is an incredibly strong statistical fit.
- MAPE (Mean Absolute Percentage Error): 0.36%: This means that the predicted value of the model is, on average, only 0.36% away from the real price. This is an extremely low percent error.

Although the RNN predictions look to be a bit smoothed and trail the actual price graphically, the very high R^2 and low error values are a strong indication that this model is highly suited to forecast the Bitcoin close price on the given test data. It follows the underlying dynamics with remarkable accuracy even though it does not precisely predict every minute fluctuation. But it must be kept in mind that past results are not always a guarantee of future performance, particularly in the highly volatile cryptocurrency market.

4.4. LSTM

Long Short-Term Memory (LSTM) networks are a particular kind of Recurrent Neural Network (RNN) used to learn long-term dependency sequences. While normal RNNs might get stuck in carrying information across many time steps because of vanishing gradients, LSTMs address this with gated memory cells that determine what to remember, forget, and to output. LSTMs are therefore particularly beneficial for time series forecasting, speech recognition, and natural language processing where the patterns span longer time spans.

In our implementation, the LSTM is given 24 hourly Bitcoin closing prices as a sequence (i.e., a complete day of hourly observations). It is trained to learn how previous hourly price changes will affect the price of the subsequent hour. The memory cells enable it to remember and use longer-range patterns in the data that can affect future values. Each training sample is a 24-hour sequence (input) and the subsequent hour's price (output). During prediction, it forecasts the subsequent hour's price based on new sequences of the previous 24 hours.

4.4.1. Why LSTM?

- Improved Memory Retention: Unlike simple RNNs, LSTM is able to recall long-term dependencies, and hence is more efficient in remembering intricate patterns in time series
- Prevents Vanishing Gradients: Due to its internal gate, it can learn faster on longer sequences.
- Handles Noise and Volatility: Financial data like Bitcoin prices are noisy and volatile—LSTMs are robust to such fluctuations and can learn meaningful signals.
- Sequential Context: LSTM has sequential data processing, learning from how past moves in prices determine future prices.

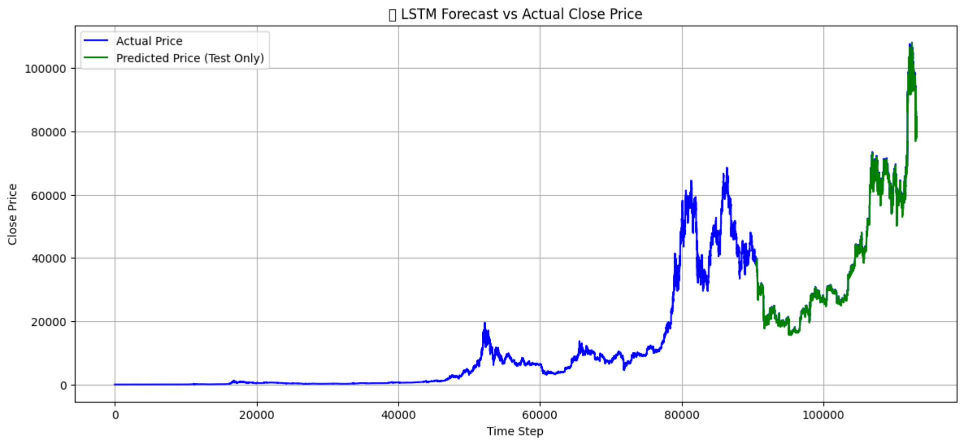


Figure 22: LSTM Forecast

From the visual perspective, the predicted price (in green) is very similar to the actual price (in blue) throughout the test period. The model appears to model both broad trends and a number of short-term fluctuations accurately, more so than the previous RNN model. Furthermore, the LSTM model-generated predictions respond faster to dramatic spikes in the direction of the upward and downward movements of the actual price, indicating less lag. Additionally, the forecasted price appears to accurately map the size of the actual price movements, indicating the LSTM is more competent in regards to volatility modeling.

4.4.2. Evaluation Metrics

- RMSE (Root Mean Squared Error): 499.3390: This is the average size of the prediction errors. Although bigger than the former RNN's RMSE, it's quite small compared to the size of the Bitcoin prices.

- R^2 Score: 0.9997: This is incredibly close to 1, which means that the LSTM model explains approximately 99.97% of the variation in Bitcoin's closing price during the testing. This is extremely high statistical correlation, close to perfection.

- MAPE (Mean Absolute Percentage Error): 0.66%: This means that the model's predictions are, on average, 0.66% off the actual price. This is an extremely small percentage difference, though slightly larger than the MAPE of the RNN.

The LSTM model possesses a very good ability to forecast the Bitcoin close price on the provided test data. Both the graphical observation of the plot and the very high evaluation metrics (R^2 nearly perfect, very low RMSE and MAPE) indicate a very accurate model.

4.5. GRU

Gated Recurrent Units (GRUs) are a particular kind of Recurrent Neural Networks (RNNs) for modeling sequential data. Like Long Short-Term Memory networks (LSTMs), they are designed to solve the vanishing gradient problem common in standard RNNs but in a more efficient architectural design. GRUs use two different gates, the reset gate and the update gate, to control information flow. GRUs are able to learn short-term and long-term dependencies in data effectively without the complexity of LSTMs under this setup.

In our case, the GRU is reading 24 hourly closing Bitcoin prices to forecast the subsequent hour's close price. It comes to realize how the past 24-hour trends and volatilities influence the subsequent hour's price. Update and reset gates help determine what parts of the history are relevant and what parts can be left behind. Every sequence is input into the GRU, and it gives an output to be verified against actual values as the model learns through changes in weights

4.5.1. Why GRU?

Easier Architecture: GRUs have fewer gates and parameters than LSTMs, so they are faster to train and less computationally costly.

Similar Performance: GRUs, being less complex, work equally well as LSTMs, particularly in situations with scarce training data.

Efficient Memory: GRUs still handle long-range dependencies effectively, making them a significant upgrade over standard RNNs.

Good Trade-Off: GRUs achieve the middle ground between performance and speed and are thus ideal for quick experimentation and real-time prediction.

4.5.2. Evaluation Metrics:

- RMSE (Root Mean Squared Error): 427.2588: This is the average size of the prediction errors. It's bigger than the earlier RNN's RMSE but smaller than the LSTM's.

- R^2 Score: 0.9996: This is very close to 1, and this means that the GRU model accounts for almost 99.96% of the test period variance of the Bitcoin close price. This is still a very good statistical fit.

- The Mean Absolute Percentage Error (MAPE) is 0.9%. This is the figure which indicates, on average, the model's prediction errors from the true price are 0.9%. Significantly, this is the largest MAPE figure seen to date among the three models, yet still relatively low.

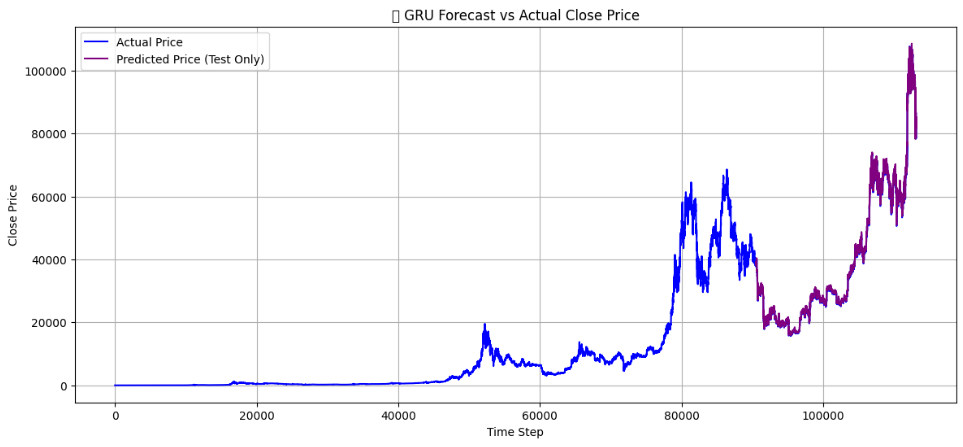


Figure 23: GRU Forecast

The GRU model demonstrates extremely high ability to forecast the Bitcoin close price on the test set, albeit maybe not quite as high as the RNN and LSTM by the metrics. The R^2 value is still extremely high, since the model accounts for the overwhelming majority of the price variation.

4.6. Multi-Feature GRU

The GRU is trained on multivariate sequences; it is given 24-hour blocks of three features: Close, Open, and a 30-hour Moving Average (MA30). It is trained to use the interaction between the three indicators in such a manner as to predict only the future Close price. The architecture of the GRU itself doesn't change (50 units, tanh activation), but it now operates on richer, more complex input data (3 features per timestep instead of 1). By including more features like Open and MA30, the model gains access to trend (MA30) and volatility/momentum (Close vs Open), which can be utilized to make more informed predictions. It's a step towards multivariate time series forecasting, where you're not working with prices as isolated points anymore but as part of a more robust temporal structure. But it also adds more noise if the extra features are not informative or if the model overfits.

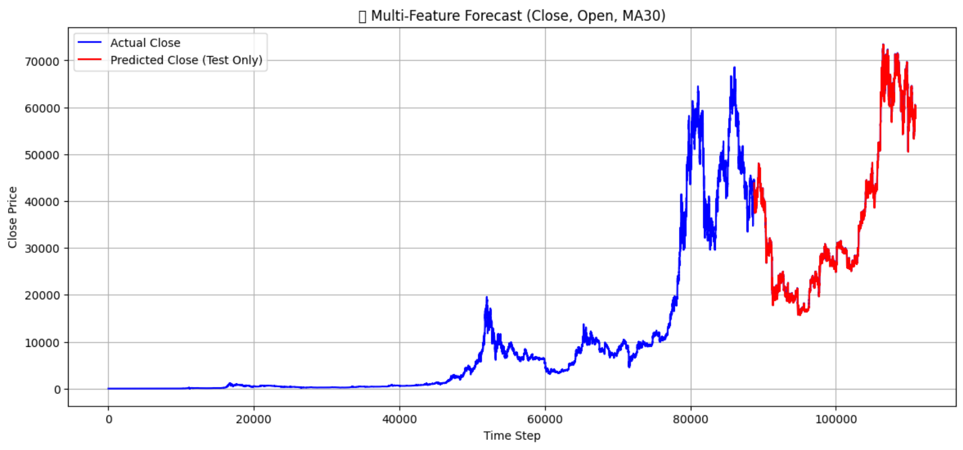


Figure 24: Multi Feature GRU Forecast

The forecasted price (red) tracks the overall patterns of upward and downward price movement in the actual closing price (blue) quite well throughout the testing interval. There are some large deviations, especially when the price is changing quickly and at a steep angle. The forecasted price lags the actual price during the times of strong upward or downward price movement. The model does seem to react to these price movements; however, it does not always react with the same velocity or intensity. There are some points where the model over- or underestimates the actual closing price, especially at the highs and lows.

4.6.1. Evaluation Metrics:

- RMSE (Root Mean Squared Error): 188.4685: This is a measure of the average size of the prediction error. This is the lowest RMSE among all the models considered till now, and this is the measure of the highest accuracy in terms of error size.
- R^2 Score: 0.9999: This is very close to 1, meaning that the multi-feature model accounts for nearly 99.99% of the variability in the Bitcoin close price during the test interval. This implies a nearly perfect fit statistically.
- MAPE (Mean Absolute Percentage Error): 0.33%: That is, on average, the model's predictions are straying from the actual price by a very low 0.33%. That is the lowest MAPE on record, which means the highest percentage accuracy.

The multi-feature model that comprises Close, Open, and MA30 showcases superior forecasting performance in terms of Bitcoin closing price than all other models that have been explored. The evaluation metrics expose remarkable results with the smallest RMSE and MAE, nearly perfect R^2, and the lowest MAPE. This implies that the incorporation of the 'Open' price and a 30-day moving average as input factors has significantly improved the model's performance in the forecast of the Bitcoin closing price.

4.7. Multi-Feature GRU with Hyperparameter Tuning

This GRU model incorporates a number of features (Close, Open, MA30) and employs systematic hyperparameter tuning to achieve best performance. Unlike earlier GRU models with fixed configuration, this model attempts various configurations and selects the best based on evaluation metrics.

Key Enhancements of This Model:

- Hyperparameter Tuning: It conducts an exhaustive grid search over a number of combinations of Gated Recurrent Unit (GRU) units, dropout rate, and batch sizes. It tests each configuration using the Root Mean Squared Error (RMSE) metric, ultimately deciding the configuration that results in the minimum test error.
- Regularization with Dropout: A Dropout layer is added to minimize overfitting through selectively turning off a percentage of units at training time. This increases the resilience of the model, particularly when dealing with unstable financial data.
- Early Stopping: To prevent redundant training and reduce the risk of overfitting, the model uses an EarlyStopping callback. Training is halted if the validation loss is not reducing for 5 consecutive epochs.
- Model Selection Strategy: Instead of sticking to one setting, this model tries many settings and chooses the best one that performs on validation alone.
- Training Period: While training for a maximum of 100 epochs is allowed, it can be cut short based on the early stopping criterion. This is a balance between learning and generalization.
- Loss Visualization: Loss curves for every configuration are graphed to provide information on convergence behavior, overfitting trends, and model stability.

Analysis of the Top Three Loss Visualizations: Number of Units Matters:

- 50 Units: Model tends to underfit, showing limited learning capacity.

- 100 Units: Generates a more potent learning signal but more prone to overfitting with further training.
- 64 Units: Achieves an equilibrium between the two—acquiring adequate learning while avoiding rapid overfitting.

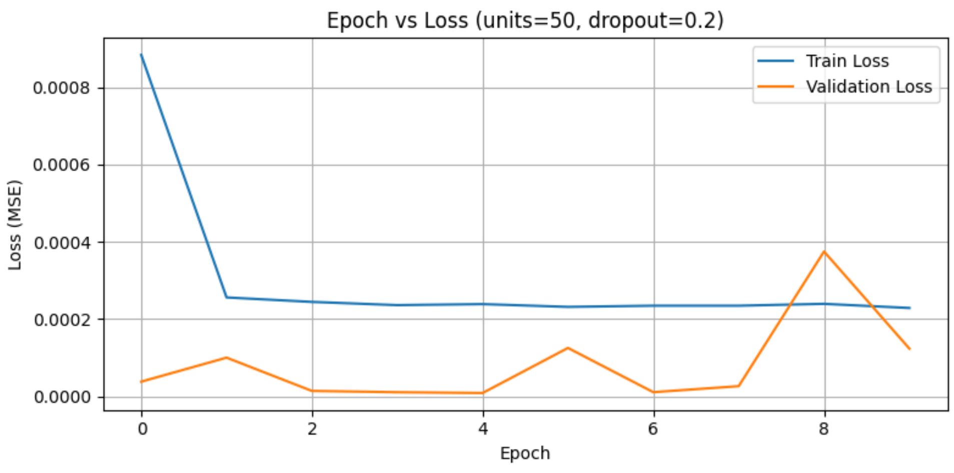


Figure 25: Epoch VS Loss (unit=50, Dropout=0.2)

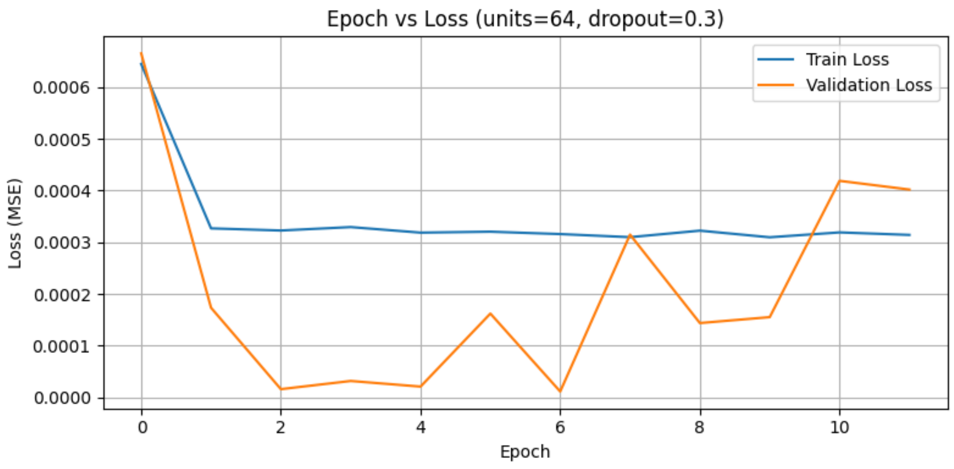


Figure 26: Epoch VS Loss (unit=64, Dropout=0.3)

These plots highlight the need to vary the number of GRU units, the value of dropout, and the number of epochs to create a proper trade-off between overfitting and underfitting. Figure - Predictive Visualization (Multi-Feature GRU) Despite some visual improvements and brief delays, the model accurately captures the overall trend and variations of Bitcoin's closing price during the testing process. It successfully absorbs information from the interrelation of different input parameters: Close, Open, and the moving average (MA30). The selected setup (presumably consisting of around 64 units, a dropout rate of 0.2, and early stopping between 40 and 60 epochs) results in a model with strong generalization power in terms of unseen data.

The improved multi-feature GRU model, optimized by hyperparameter tuning and regularization, offers a robust and accurate forecasting method. The combination of multiple features, careful parameter tuning, as well as training procedures like dropout and early stopping enable it to overcome the issues inherent in both underfitting and overfitting. Even with the occurrence of certain smoothing in the

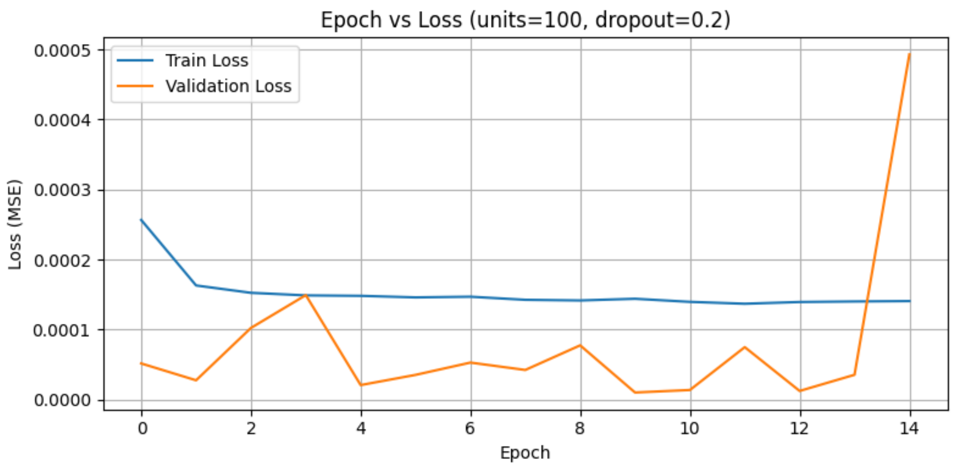


Figure 27: Epoch VS Loss (unit=100, Dropout=0.2)

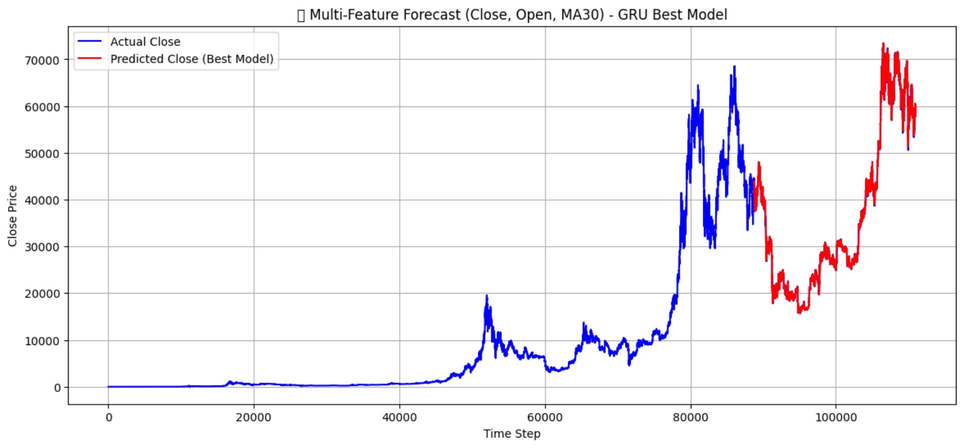


Figure 28: Multi Feature Forecast

forecasts, overall performance and generalization capacity remain good, given the inherent volatility that is inherent in cryptocurrency data.

5. Final Results And Conclusion

Model Performance Comparison (Lower is Better for RMSE and MAE):

- SARIMA: Indicates the largest discrepancies, with an RMSE of around 7501.9 and an MAE of 6589.1. This indicates that it was the least accurate of the models under comparison.
- SARIMAX: Indicates a vast improvement compared to SARIMA, RMSE 2458.5 and MAE 1640.3. Incorporating exogenous variables assisted in the improvement of the model.
- RNN: Exhibits extremely small errors compared to SARIMA and SARIMAX, with RMSE = 323.3 and MAE = 161.0. This indicates a very large improvement in accuracy with the application of a recurrent neural network.
- LSTM: Slightly higher errors than the RNN but still significantly better than SARIMA and SARIMAX, with an RMSE of 409.3 and an MAE of 275.6. The Long Short-Term Memory network also performed very well.

- GRU: Exhibits comparable errors to the LSTM with RMSE = 427.3 and MAE = 349.7. Gated Recurrent Unit was also very good at prediction.
- GRU+Open+MA30: Has the lowest errors among all the models, with an RMSE of 188.5 and an MAE of 118.3. This shows that the addition of the 'Open' price and a 30-day Moving Average as features greatly enhanced the performance of the GRU model.

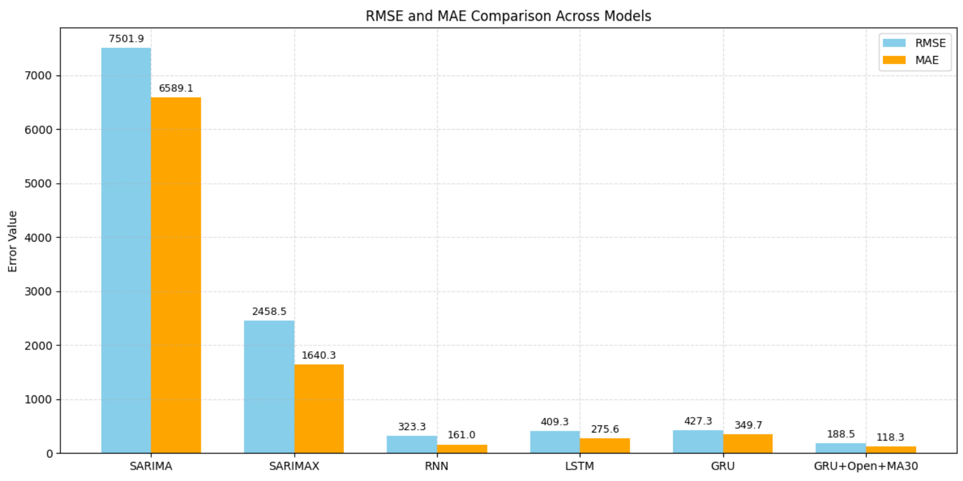


Figure 29: RMSE and MAE Comparisions

The GRU model with incorporation of the 'Open' price with a 30-day Moving Average (labeled as GRU+Open+MA30) outperformed both RMSE and MAE, indicating the highest accuracy in forecasting the closing price of Bitcoin among the models. The recurrent neural network models like RNN, LSTM, and GRU outperformed the traditional time series models like SARIMA and SARIMAX, and inclusion of relevant exogenous variables greatly enhanced the predictive capability of the GRU model.