Zelta Labs
Algorithmic Trading Model Development for BTC/USDT Crypto Market

Final Report

Team Ctrl+Alt+Defeat

Aditya Jha
Somyajeet Chowdhury
Jagori Bandyopadhyay
Shristi Singh

Kharagpur Data Science Hackathon 2024

1. INTRODUCTION

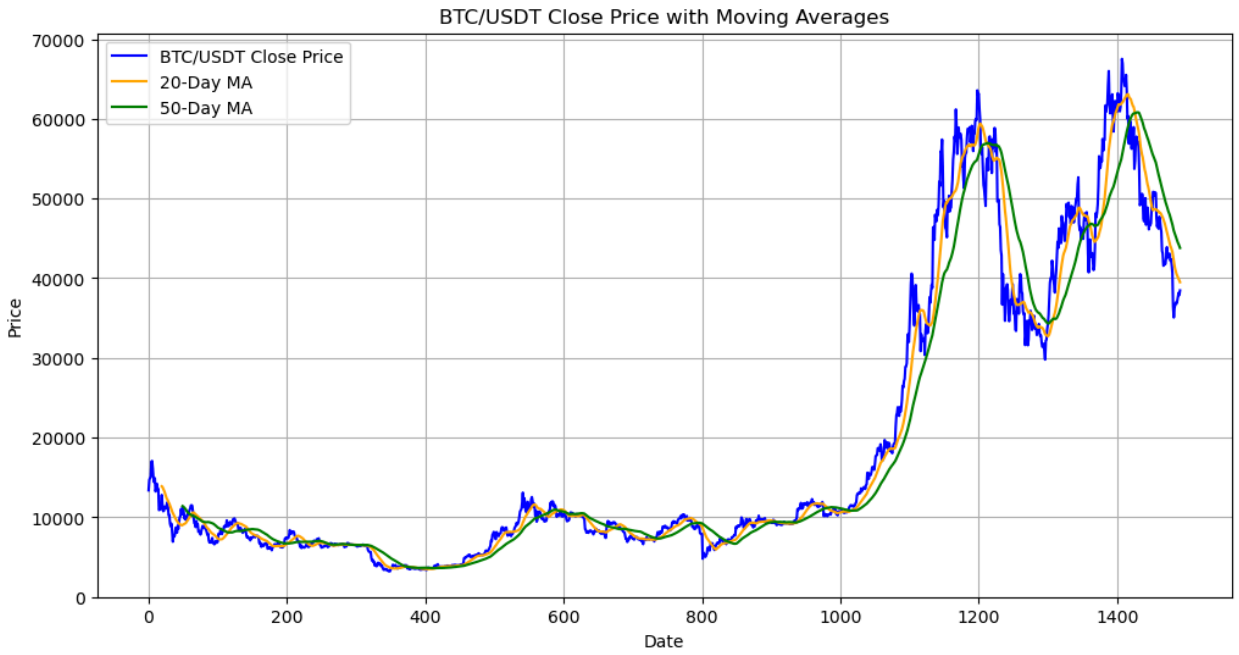
Embarking on our expedition into the dynamic BTC/USDT cryptocurrency market demands a strategic approach, akin to navigating uncharted territories. This realm of algorithmic trading operates within the swift currents of opportunity and risk, requiring our constant vigilance.

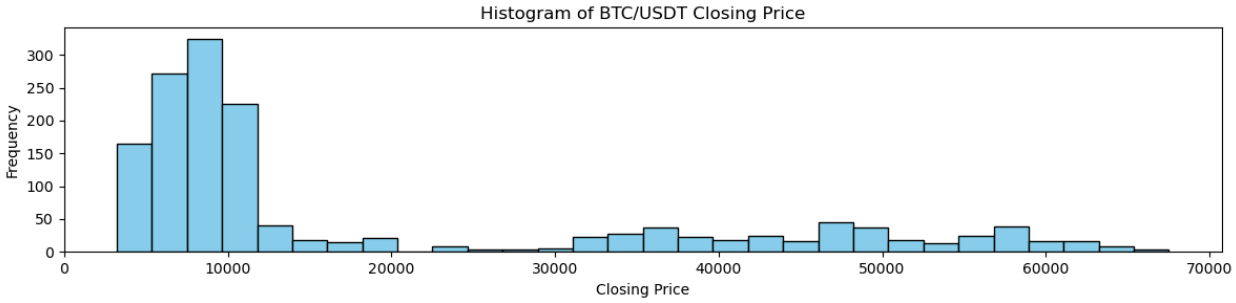
As active contributors to this intricate landscape, we embrace the multifaceted challenges, blending innovation and strategic foresight. Our focus extends beyond mere market resilience; our meticulously crafted trading techniques not only overcome unpredictable market fluctuations but surpass established benchmarks.

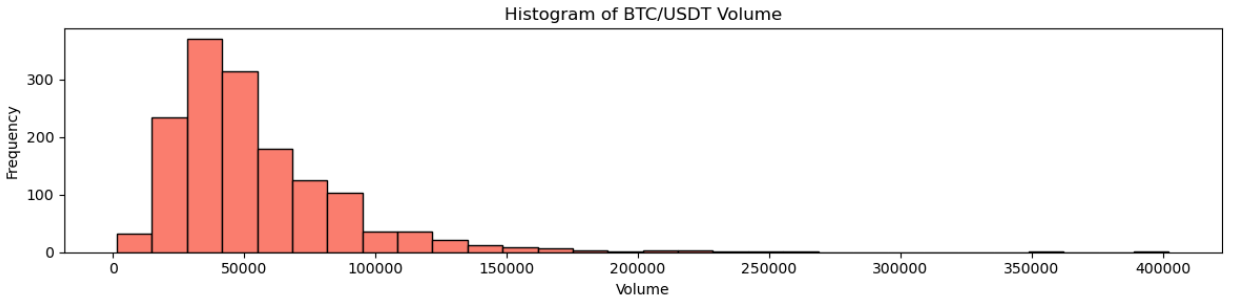
Our collective efforts converge into a sophisticated amalgamation of algorithmic precision and risk management. We meticulously design and deploy algorithms tailored to navigate the nuances of the BTC/USDT market, ensuring a delicate balance between returns and risk mitigation.

Cryptocurrencies, by their nature, are in a perpetual state of flux. Our adaptability is a cornerstone, necessitating a keen awareness of prevailing market trends. We intend to offer insights that stimulate fresh ideas, foster collaboration, and advocate for methodologies where algorithms stand as pivotal tools in maximizing returns and controlling risks.

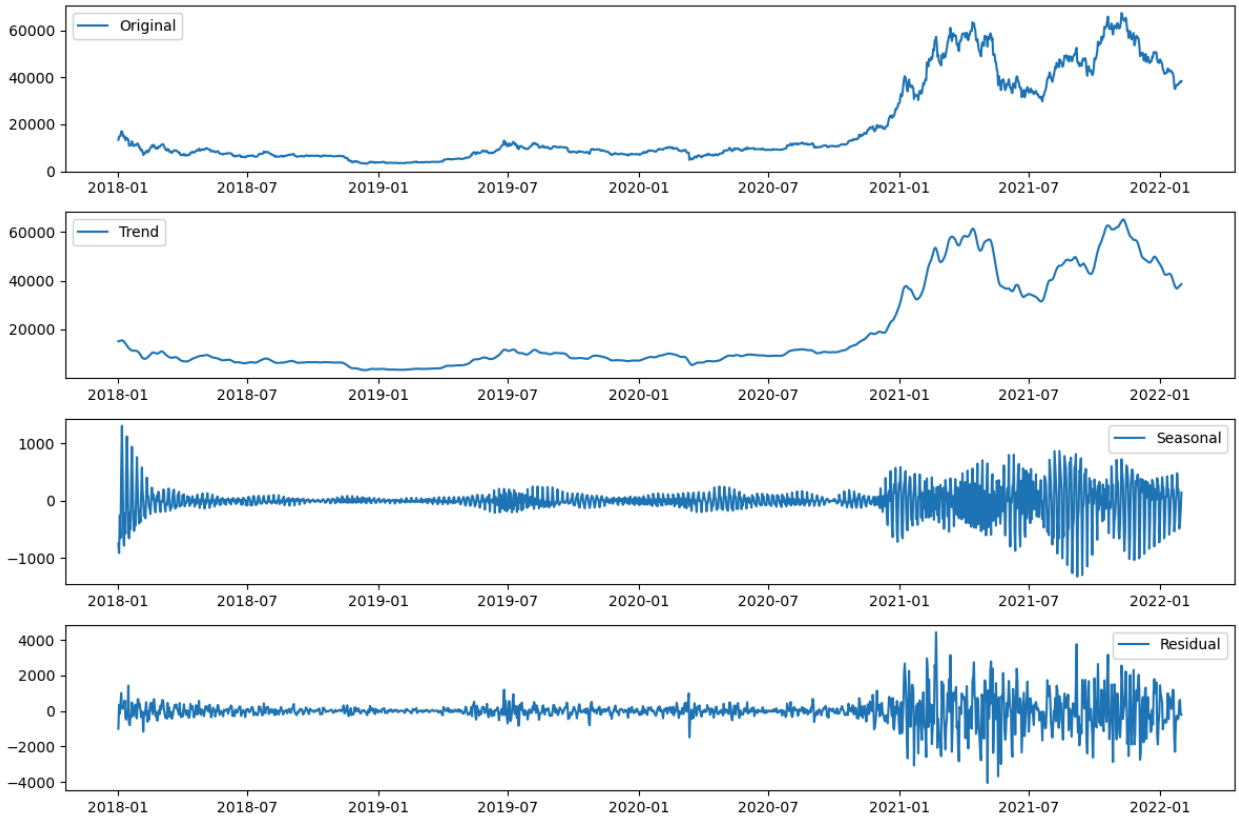
2. EXPLORATORY DATA ANALYSIS

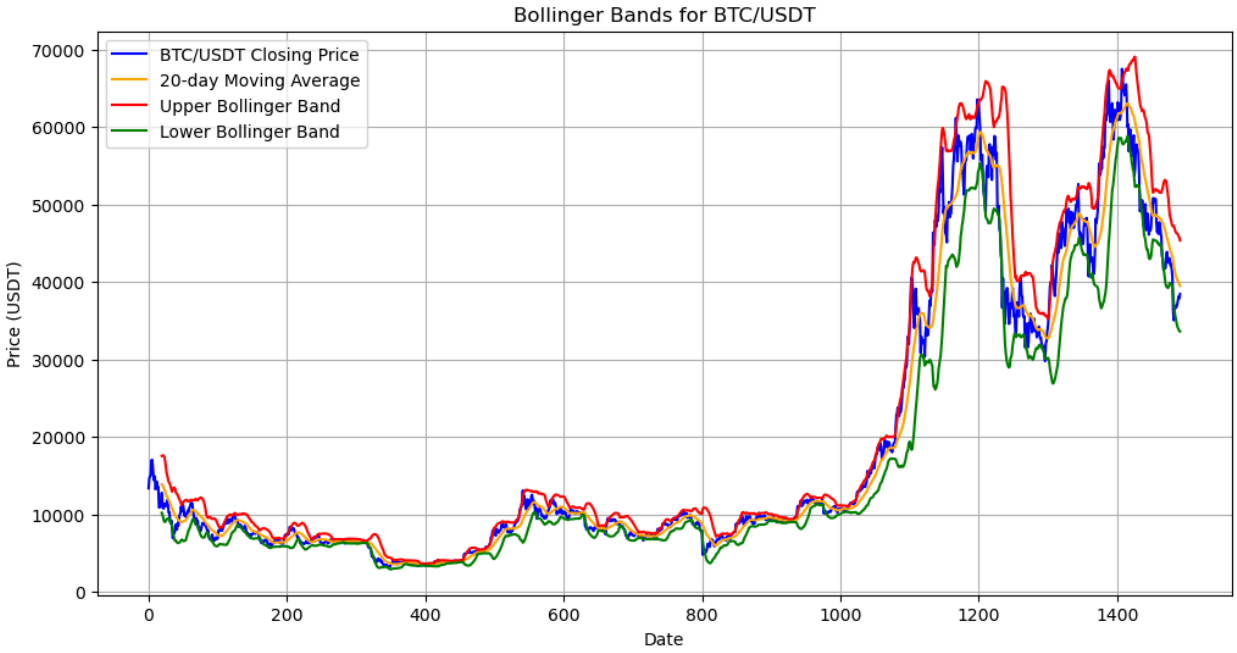


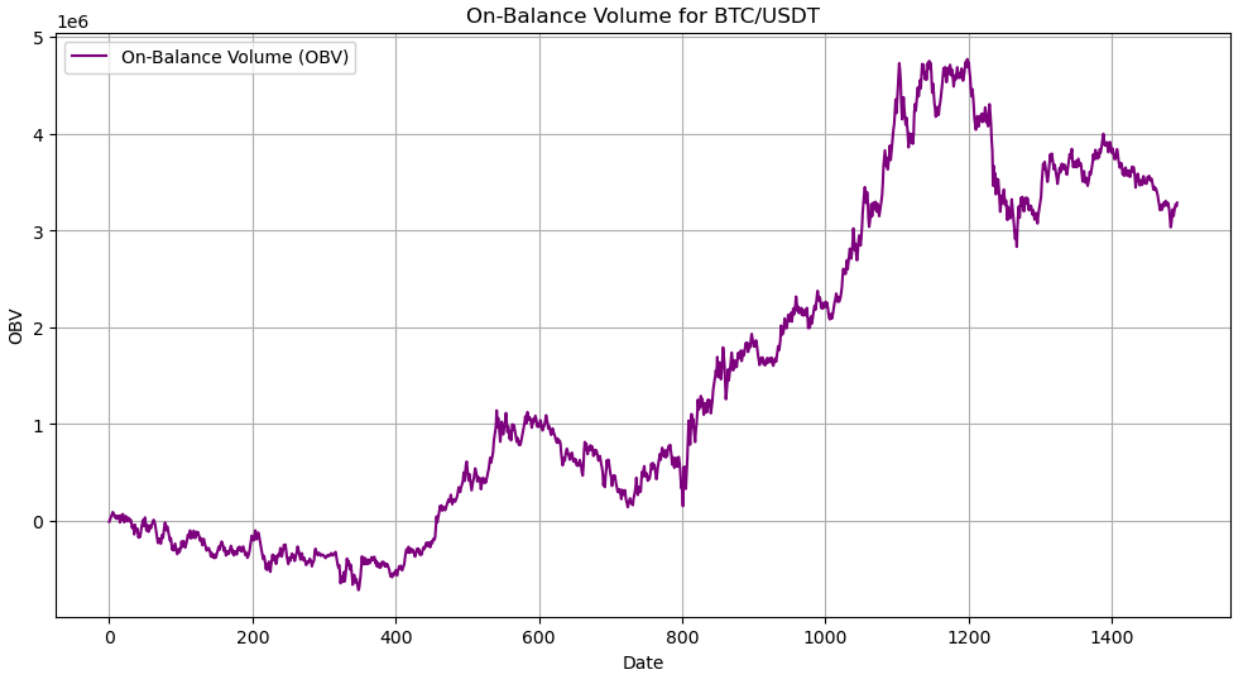


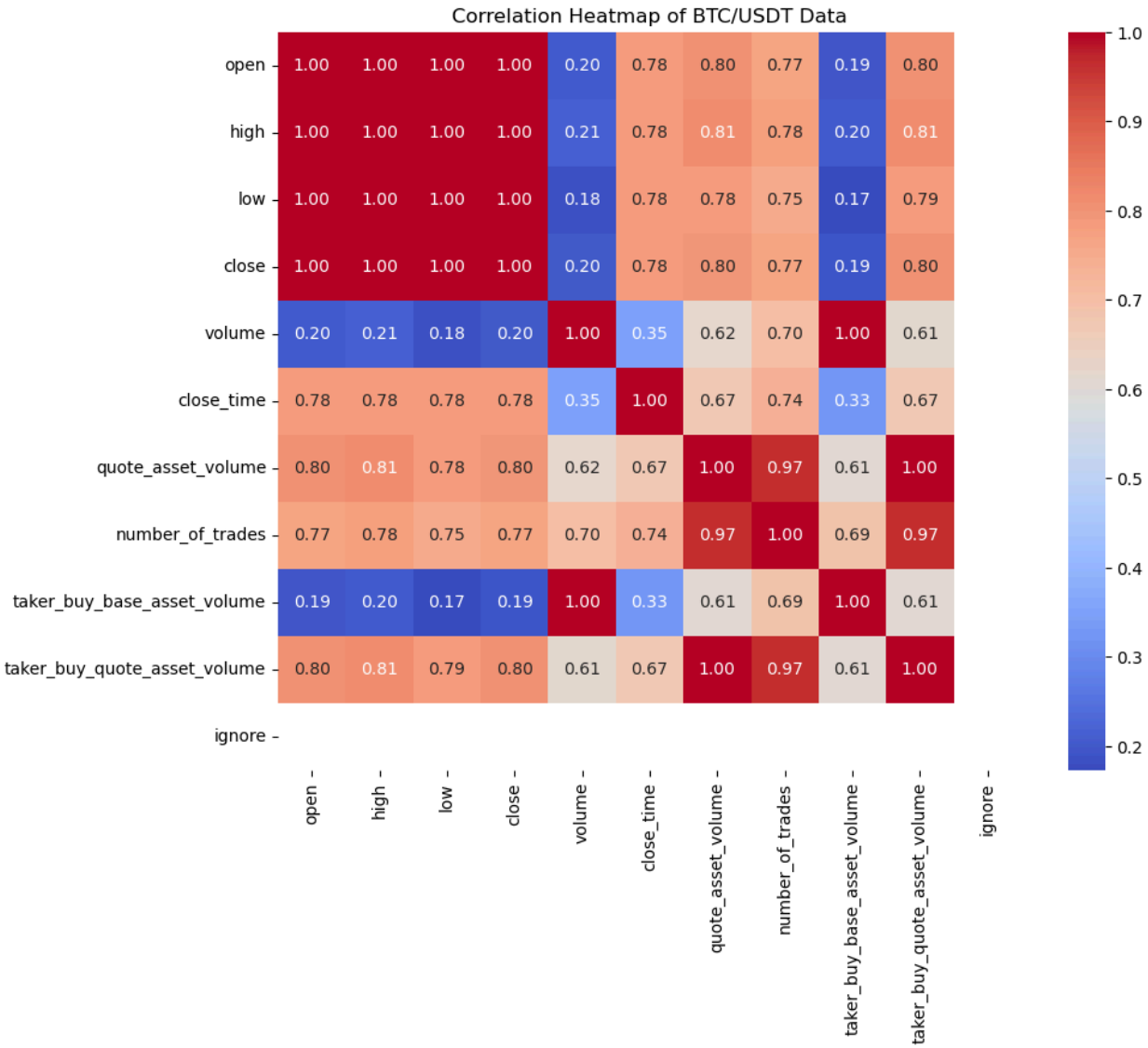


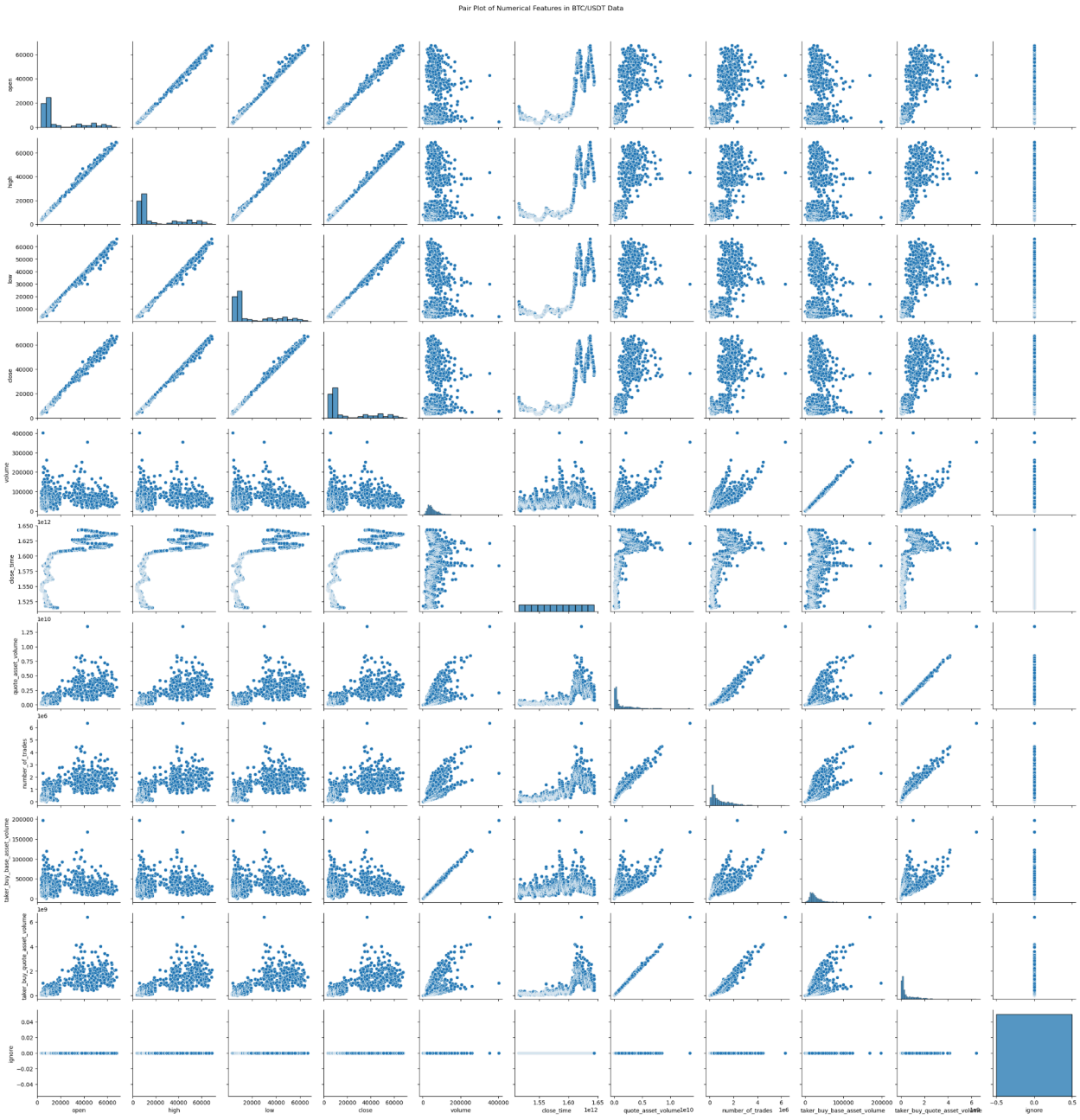
STL ANALYSIS



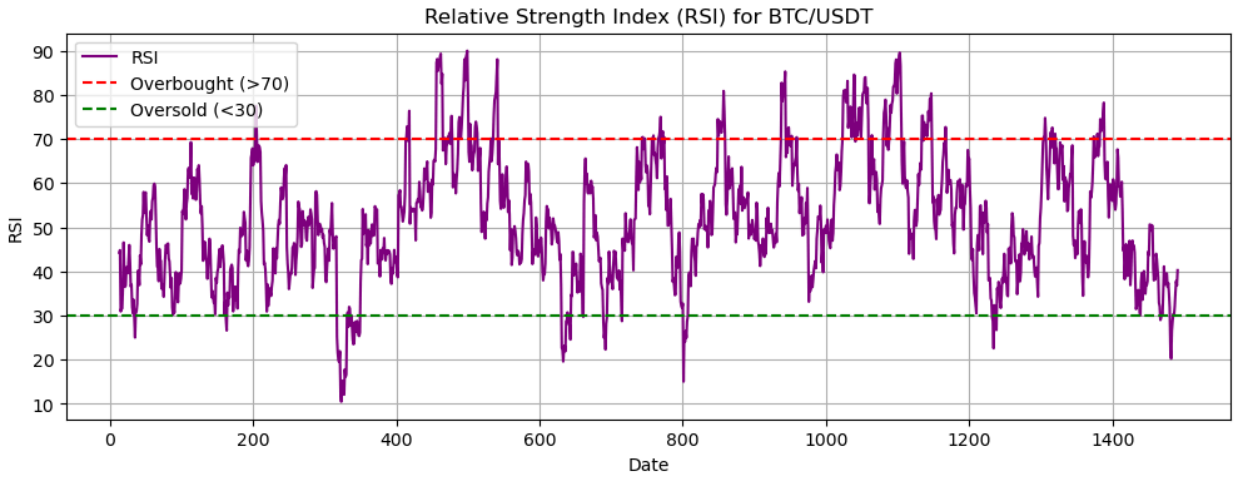

Bollinger Bands for BTC/USDT
BTC/USDT Closing Price
20-day Moving Average
Upper Bollinger Band
Lower Bollinger Band
Price (USDT)
Date
0
10000
20000
30000
40000
50000
60000
70000
0
200
400
600
800
1000
1200
1400

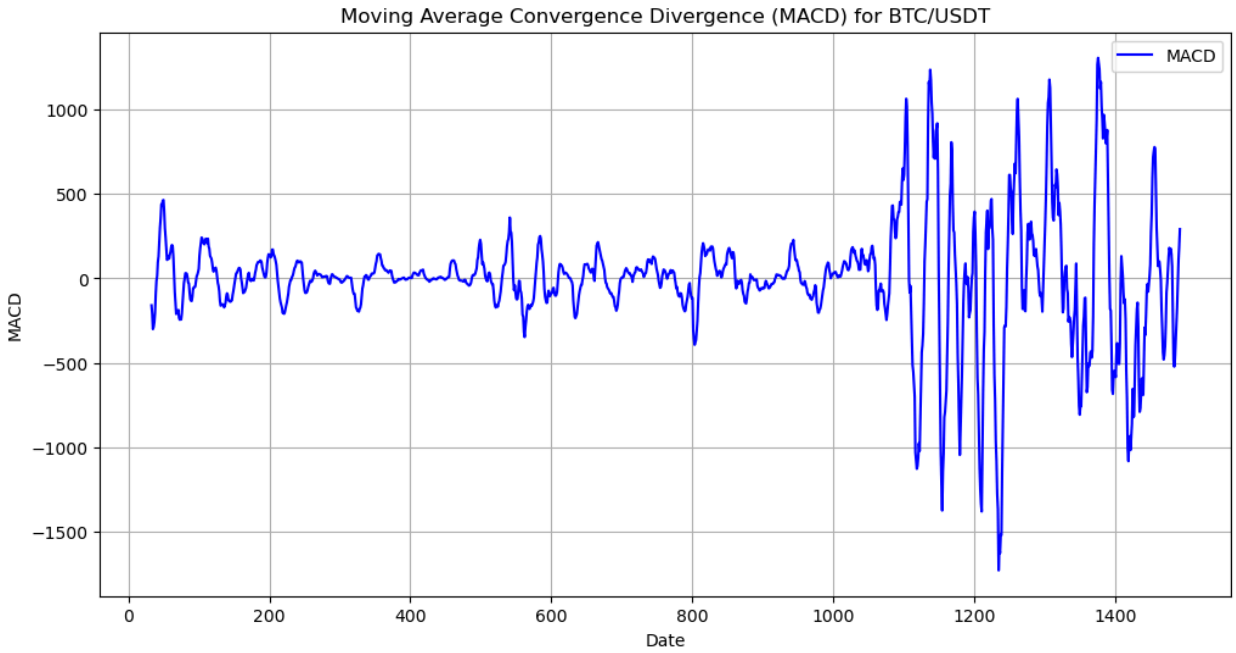

On-Balance Volume for BTC/USDT
1e6
On-Balance Volume (OBV)
OBV
Date
0
1
2
3
4
5
0
200
400
600
800
1000
1200
1400

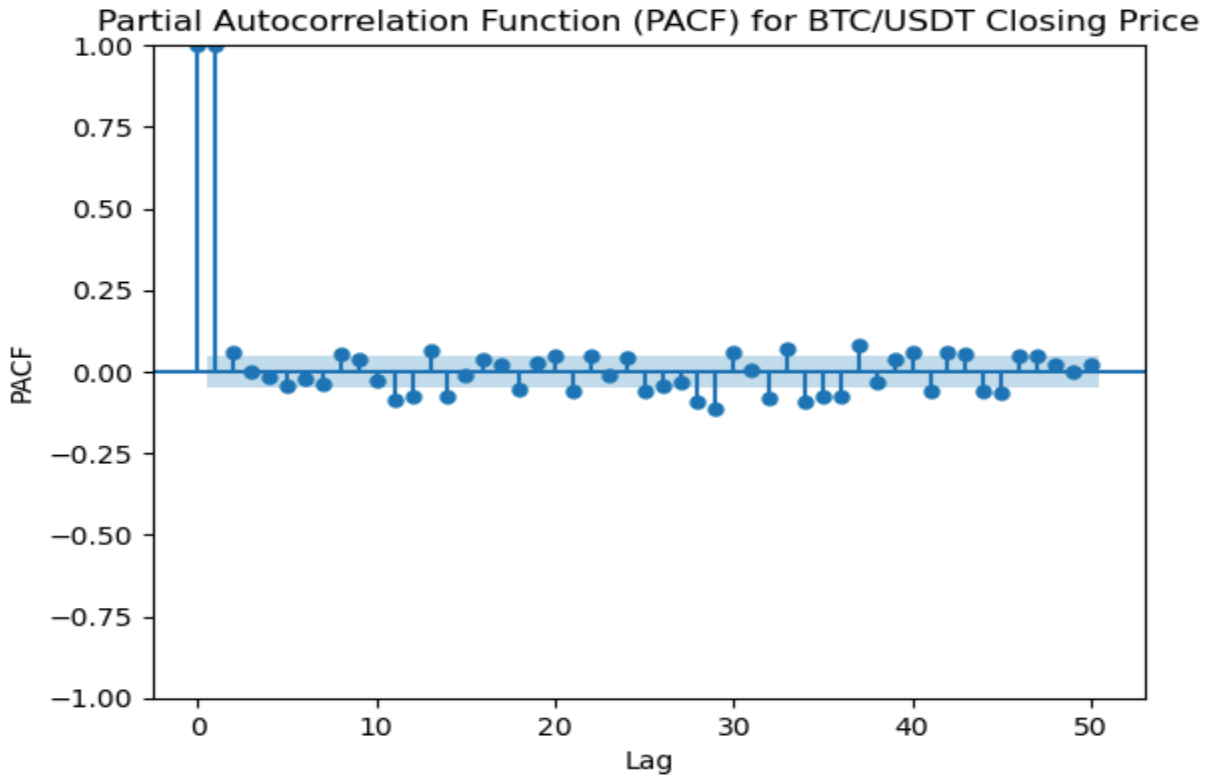

Correlation Heatmap of BTC/USDT Data
open 1.00 1.00 1.00 1.00 0.20 0.78 0.80 0.77 0.19 0.80
high 1.00 1.00 1.00 1.00 0.21 0.78 0.81 0.78 0.20 0.81
low 1.00 1.00 1.00 1.00 0.18 0.78 0.78 0.75 0.17 0.79
close 1.00 1.00 1.00 1.00 0.20 0.78 0.80 0.77 0.19 0.80
volume 0.20 0.21 0.18 0.20 1.00 0.35 0.62 0.70 1.00 0.61
close_time 0.78 0.78 0.78 0.78 0.35 1.00 0.67 0.74 0.33 0.67
quote_asset_volume 0.80 0.81 0.78 0.80 0.62 0.67 1.00 0.97 0.61 1.00
number_of_trades 0.77 0.78 0.75 0.77 0.70 0.74 0.97 1.00 0.69 0.97
taker_buy_base_asset_volume 0.19 0.20 0.17 0.19 1.00 0.33 0.61 0.69 1.00 0.61
taker_buy_quote_asset_volume 0.80 0.81 0.79 0.80 0.61 0.67 1.00 0.97 0.61 1.00
ignore
open
high
low
close
volume
close_time
quote_asset_volume
number_of_trades
taker_buy_base_asset_volume
taker_buy_quote_asset_volume
ignore
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2



PAIR PLOT CALCULATION


Relative Strength Index (RSI) for BTC/USDT
RSI
Overbought (>70)
Oversold (<30)
RSI
Date
90
80
70
60
50
40
30
20
10
0
200
400
600
800
1000
1200
1400

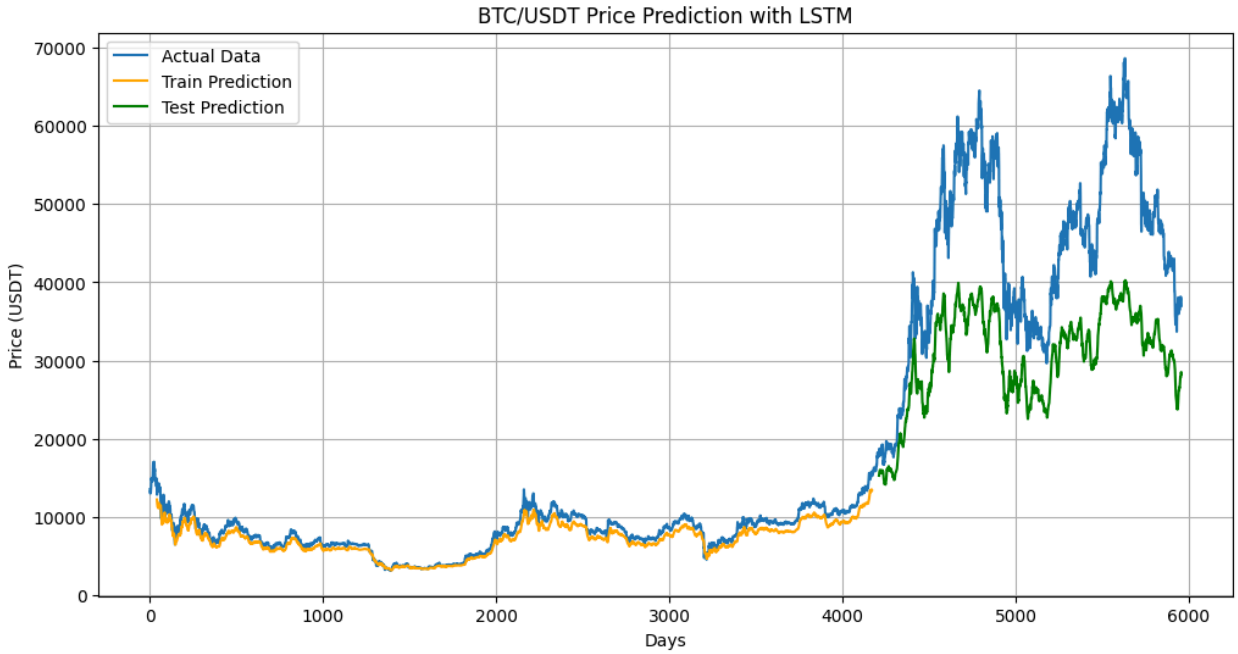

Moving Average Convergence Divergence (MACD) for BTC/USDT
MACD
MACD
Date
1000
500
0
−500
−1000
−1500
0
200
400
600
800
1000
1200
1400



3. DEVELOPING PREDICTIVE MODEL

a. LSTM

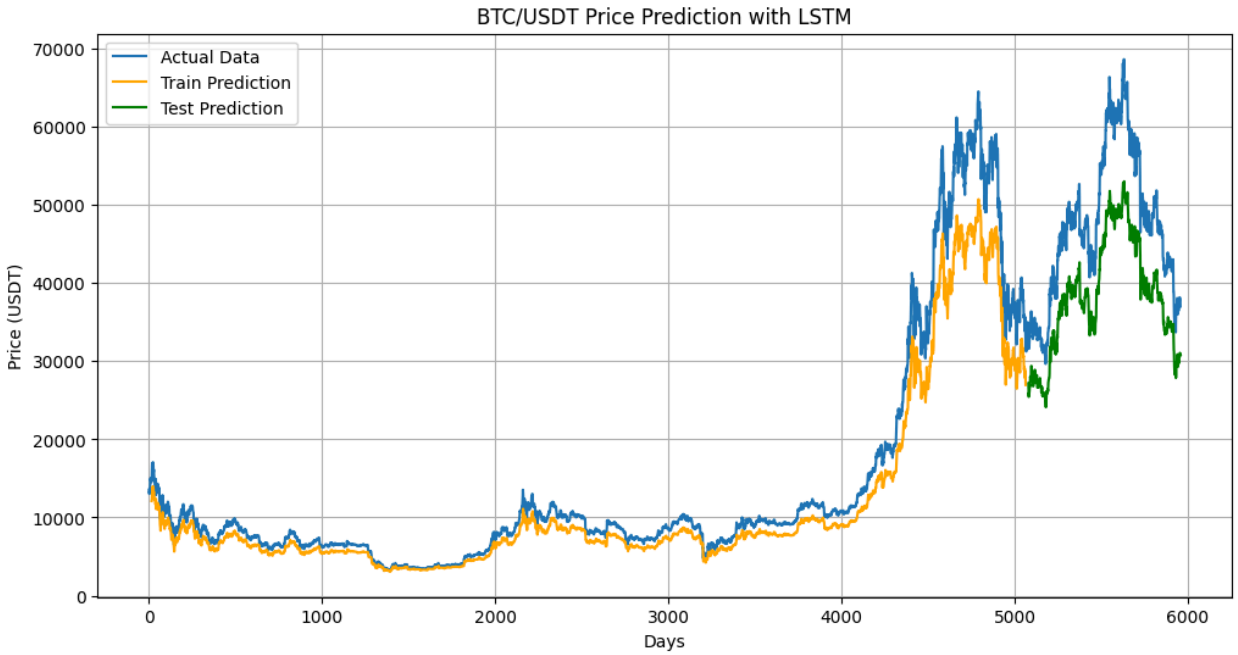
We first started by trying to predict the future trends (closing prices) of the stock market, so we implemented a basic LSTM model with the first layer having 4 memory units. This layer is designed to capture long-term dependencies in sequential data. After the LSTM layer, there is a Dense layer with 1 unit. This layer is a fully connected layer that outputs a single value. A Dropout layer with a dropout rate of 0.2 is added to ensure regularization to help prevent overfitting. The model is trained on 70 % of 6-hour data using mean squared error loss, and early stopping is employed to prevent overfitting and select the best-performing model. The model scored 1007 RMSE on the training dataset and 15091.13 RMSE on the test dataset.



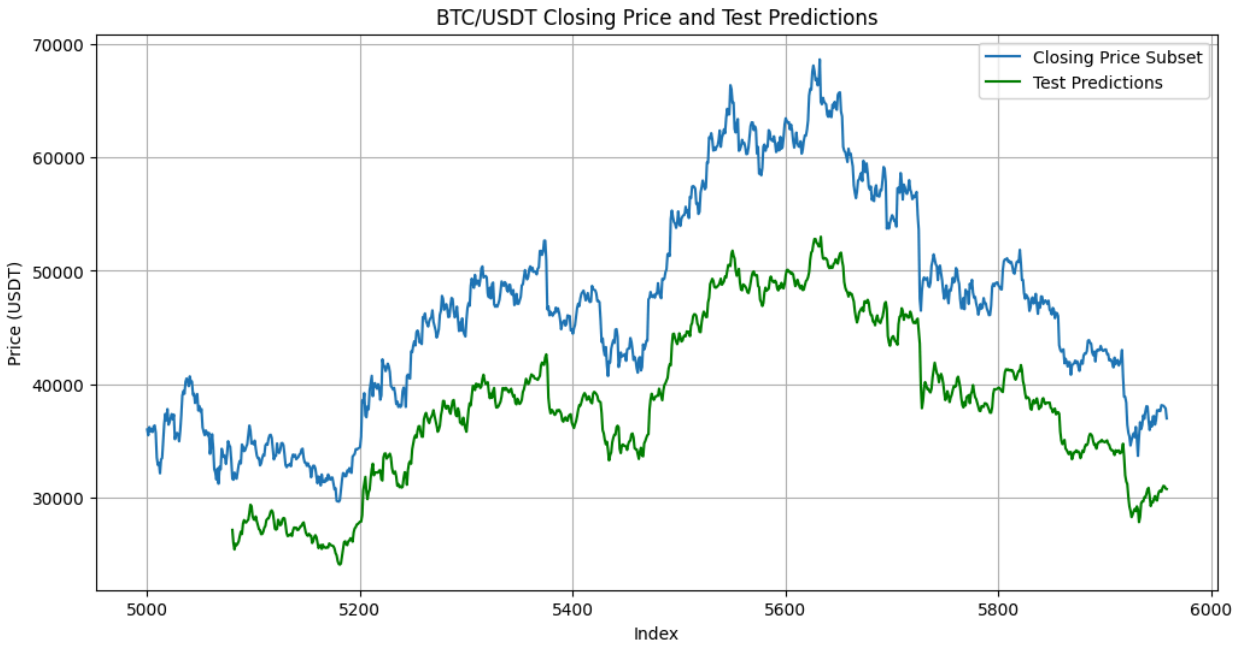
b. LSTM - Modified

We observed that the trend and general direction of price movement that we predicted was spot on but the peaks of the prediction did not match.
We observed that the data on which we trained the model was bounded to lower price levels and the market experienced a boom just after the training data was exhausted the magnitude of which it was unable to predict accurately.

To overcome this challenge, we decided to expose the model to more of the dataset while training so that it is more equipped to predict such market booms.

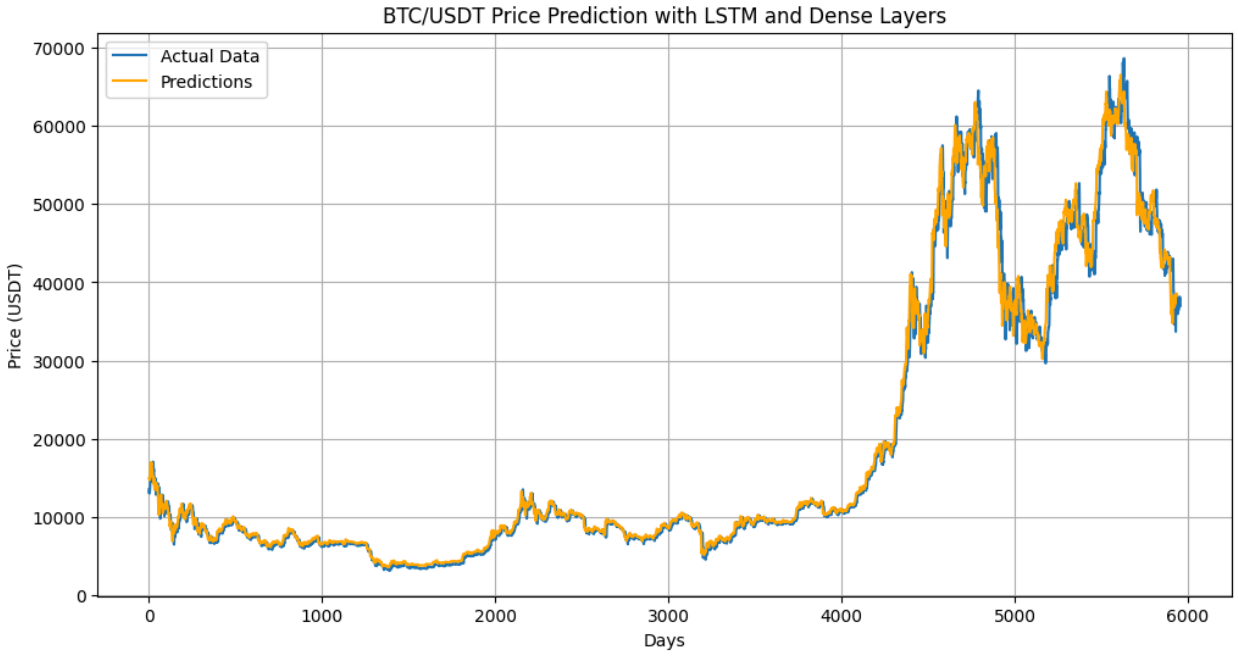


We divided the 6-hour data into an 80-20 ratio for training and testing, respectively. We achieved a train score of 3692.30 RMSE and a Test Score of 9439.76 RMSE, achieving a 38% decrease in the value from the initial model.
The following is the zoomed-in version where we focus only on the test predictions. We can still see a prominent gap between the actual and predicted data



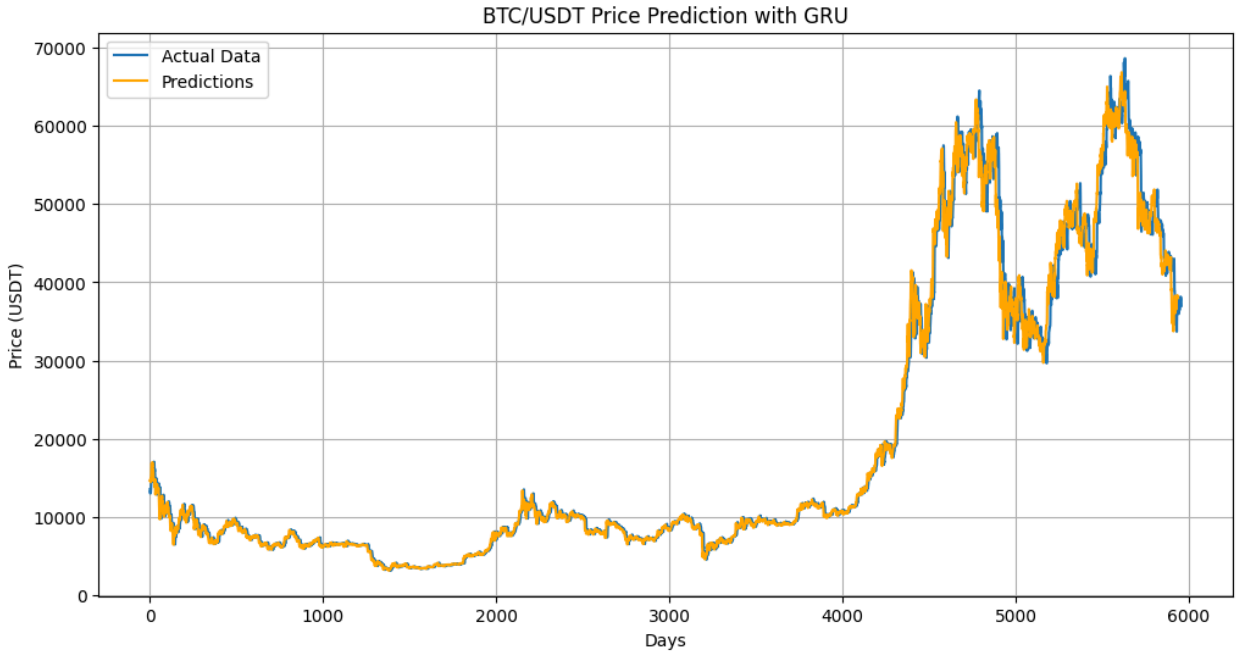
c. LSTM with Dense Layers

To further improve the prediction, we deployed more dense layers to improve the accuracy. So in this model we had a dense layer with 25 nodes before the final dense layer to output a final prediction. We achieved a train score of 456.30 RMSE and a test score of 1114.08 RMSE. This reduced the RMSE further by a whopping 88%



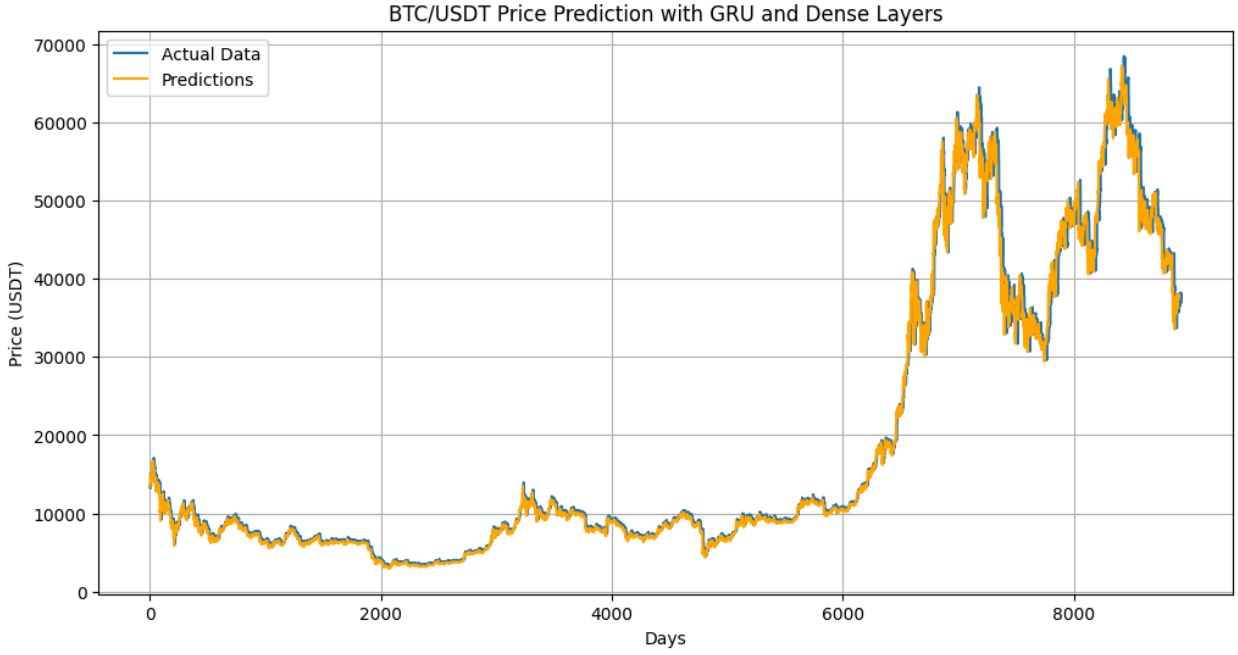
d. GRU

Because the model worked comparatively well on LSTM, which is a type of RNN, we expected the model to work well when trained through GRU. When applied, GRU did even better than the prior LSTM models, producing a test score of 976.50 RMSE



e. GRU with Dense Layer

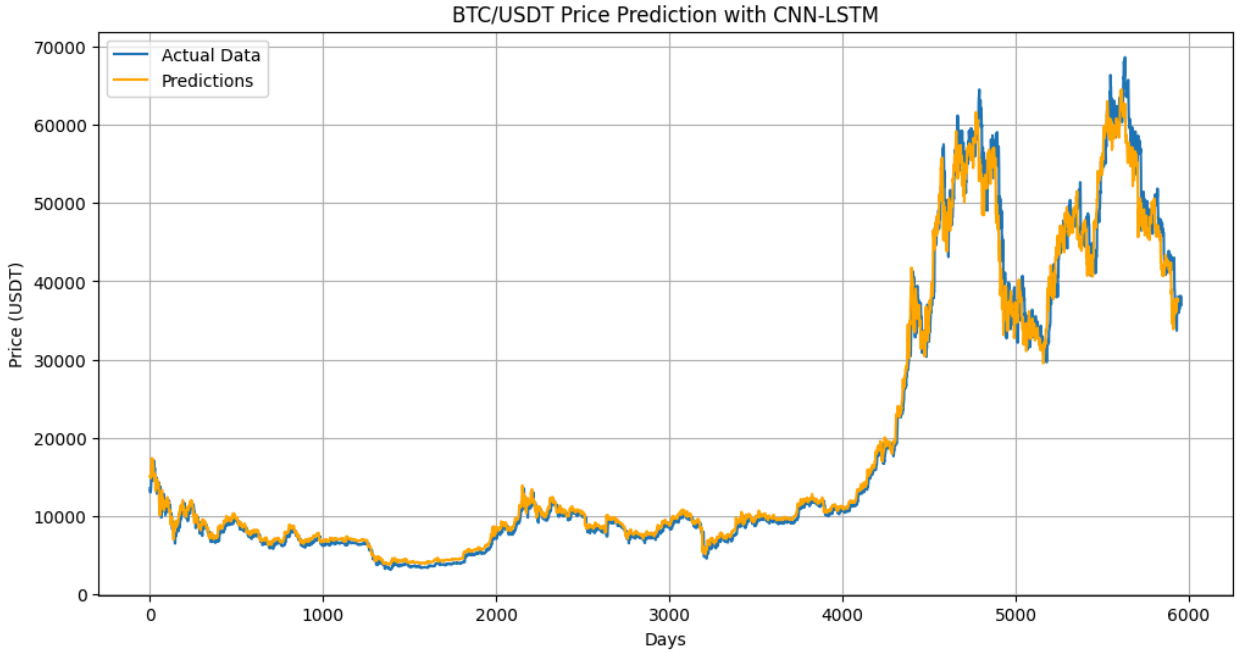
In an attempt to increase the accuracy of GRUs, we added a dense layer in its architecture with 50 nodes, and marginally decreased losses in the test dataset to 857.17 RMSE



f. CNN and LSTM

Before finalising our model we just wanted to try how this dataset performed when combined with CNN. CNN seemed intuitive because of how CNN creates a feature map by sliding a filter over the input matrix and its ability to represent data in a large dataset.

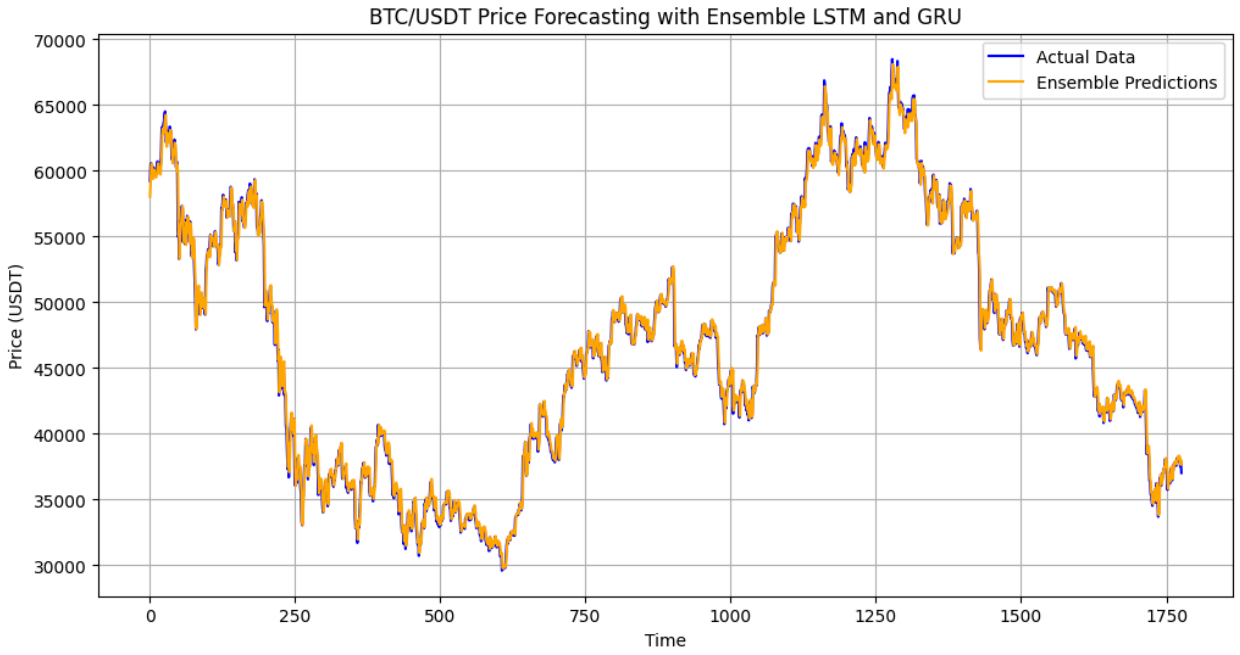
Upon combining CNN and LSTM, we achieved a slightly worse RSME of 1476.70 on test data



g. LSTM-GRU Ensemble

Finally, we decided to combine LSTM and GRU through a bagging ensemble in order to receive the advantages of the simplicity of GRU as well as the LSTMs long-term memory-retaining capabilities. The performance is described in the table below:

	6 hrs	4 hrs	1 hr
LSTM test score	1006.63	808.78	384.11
GRU test score	923.06	793.35	395.88
Ensemble test score	942.91	779.07	389.19



4. EXPLAINING TRADING STRATEGIES AND OUR CODE

a. Indicators:

i. Simple Moving Averages (SMA):

- **Support and Resistance:** BTC price movements can interact with SMAs, serving as dynamic support or resistance levels. Traders often look for opportunities when the BTC price interacts with these moving averages.
- **Trend Identification:** SMAs are valuable for trend identification, which is crucial in the BTC market where sustained trends can provide profitable opportunities.
- has context menu
- `sma_period_short` and `sma_period_long` define the periods for short and long SMAs, respectively. The short SMA measures the SMA of 50 days, and the long SMA one measures the SMA of 200 days.
- Two SMAs are used to identify trends.

ii. Relative Strength Index (RSI):

- **Volatility Management**: The cryptocurrency market, including BTC, can be highly volatile. RSI helps traders identify overbought and oversold conditions, allowing for more informed decisions on managing volatility and avoiding potential market extremes.
- **Psychological Levels:** BTC trading often involves psychological levels (e.g., round numbers). RSI can be particularly useful in identifying potential reversals or exhaustion points around these levels.
- `rsi_period` determines the lookback period for RSI. it is set to 14 days
- `rsi_overbought` and `rsi_oversold` set the threshold values for overbought and oversold conditions.. Overbought is set to over 60, oversold is less than 40.

iii. **Moving Average Convergence Divergence (MACD):**

- **Trend Confirmation:** Given BTC's tendency to exhibit strong trends, MACD can be effective in confirming trend directions and signalling potential entry or exit points.
- **Cryptocurrency Market Momentum:** Cryptocurrencies are known for their momentum-driven price movements. MACD, with its focus on momentum, can provide insights into the strength of price trends in the BTC market
- `macd_short`, `macd_long`, and `macd_signal` set the periods for short-term, long-term, and signal line components.

iv. **Average True Range (ATR)**

- **Volatility Measurement:** In the context of cryptocurrencies, particularly Bitcoin, where price fluctuations can be rapid and substantial, ATR provides quantifiable insights into market volatility. High ATR values indicate periods of increased price volatility, allowing traders and investors to adapt their strategies accordingly.
- **Dynamic Risk Management:** In the cryptocurrency market, it assists traders in setting adaptive stop-loss and take-profit levels based on the current volatility, enabling a more responsive approach to the dynamic nature of Bitcoin's price movements.
- `atr_period` sets the lookback period for ATR

v. **Average Directional Movement Index (ADX)**

- **Adapting to Changing Trends:** BTC often experiences rapid changes in trend direction. ADX helps traders adapt to changing market conditions by providing a measure of trend strength, indicating whether trends are developing or losing momentum.
- **Identifying Strong Trends:** Cryptocurrency markets can exhibit strong trends, and ADX helps identify and measure the strength of these trends, assisting traders in staying on the right side of the market.
- `adx_period` defines the lookback period for ADX.

vi. **Bollinger Bands (BBand):**

- **Volatility Measurement and Price Range:** Bollinger Bands provide a visual representation of Bitcoin price volatility, with wider bands indicating higher volatility and narrower bands suggesting lower volatility. The upper and lower bands dynamically adjust to changes in market conditions, offering insights into the potential extent of price movements.
- **Trend Identification and Reversals:** Bollinger Bands are effective in identifying trends and potential trend reversals in Bitcoin prices. During uptrends, prices tend to hug the upper band, while in downtrends, they often stay close to the lower band. Touches or breaches of the bands can signal overbought or oversold conditions, providing traders with potential reversal points.
- `bband_period` sets the lookback period for Bollinger Bands
- `bband_dev` determines the standard deviation multiplier.

b. Implementation of Strategy

i. **Introduction:**

- The strategy's main logic is implemented in the `next` method, which is called for each new data point.
- The strategy checks if there is enough historical data (`len(self.data) > self.params.sma_period_long`)

ii. Buy Conditions:

- If the closing price is above both short and long SMAs.
- RSI is below 45 (indicating potential oversold condition).
- MACD is positive
- ADX is above (indicating a strong trend).
- No open position exists.

iii. Sell Conditions:

- If the closing price is below both short and long SMAs.
- RSI is above 55 (indicating potential overbought condition).
- MACD is negative.
- ADX is above (indicating a strong trend)
- An open position exists.

5. BACKTESTING SETUP:

a. Data:

- Bitcoin (BTC/USDT) 5-minute OHLCV data is loaded from a CSV file.

b. Broker and Analyzer Setup:

- Initial capital is set to $10,000.
- A commission of 0.1% is applied to each trade.
- Analyzers, including Sharpe Ratio, TradeAnalyzer, and DrawDown, are added to evaluate performance.

c. Execution and Results:

- The strategy is run using the `cerebro.run()` method.
- Starting and ending portfolio values are printed.
- The code includes commented-out sections for additional performance metrics such as total closed trades, win rate, and maximum drawdown.

6. RISK MANAGEMENT:

a. Overall Benefits for Risk Management:

- **Defined Risk-Reward Ratio:** By setting stop-loss and take-profit levels, traders can establish a clear risk-reward ratio for each trade. This ratio helps ensure that potential losses are controlled and that profits are secured at predefined levels.
- **Consistency in Trading Approach:** Stop loss and take profit orders contribute to a consistent and disciplined trading approach. Traders can stick to their predetermined risk and reward levels, avoiding impulsive decision-making.

- **Mitigation of Emotional Factors:** Automated exit strategies reduce the impact of emotions on trading decisions. Fear and greed often drive impulsive actions, and using stop-loss and take-profit orders helps mitigate these emotional influences.

b. Stop Loss:

i. Definition: A stop-loss order is set at a predetermined price level, below the current market price for a long position or above for a short position.

ii. Purpose:

- **Risk Limitation:** It serves as a safety net to limit potential losses. If the market moves against the trader, the stop-loss order automatically triggers, closing the position and preventing further losses.
- **Dynamic Adjustments:** Algorithms can dynamically adjust stop-loss levels based on market conditions, allowing for adaptive risk management.

c. Take Profit:

i. Definition: A take-profit order is set at a predetermined price level, above the current market price for a long position or below for a short position.

ii. Purpose:

- **Profit Lock-In:** It allows traders to secure profits by automatically closing a position when the market reaches a favourable price level. This helps prevent potential reversal or retracement that could erode profits.
- **Objective Decision-Making:** Similar to stop loss, take profit orders assist in making objective and systematic decisions. The algorithm can stick to its profit targets rather than succumbing to greed or indecision.

7. PERFORMANCE:

Analyzing results

In [ ]:

In [123]:
```
strategy1 = result[0]
print('Sharpe Ratio:', strategy1.analyzers.mysharpe.get_analysis()['sharperatio'])
analysis_drawdown = strategy1.analyzers.drawdown.get_analysis()
analysis_trades = strategy1.analyzers.tradeanalyzer.get_analysis()

gross_profit = analysis_trades['pnl']['gross']['total']
net_profit = analysis_trades['pnl']['net']['total']
total_closed_trades = analysis_trades['total']['closed']

print(f'Gross Profit: {gross_profit:.2f}')
print(f'Net Profit: {net_profit:.2f}')
print(f'Total Closed Trades: {total_closed_trades}')
```

```
Sharpe Ratio: 0.8227504996334666
Gross Profit: 36680.39
Net Profit: 30891.02
Total Closed Trades: 195
```

Actuating trade on BTC-USDT market

In [122]:
```
# Print the starting conditions
print(f'Starting Portfolio Value: {cerebro.broker.getvalue():,.2f} USDT')

result = cerebro.run()


print(f'Ending Portfolio Value: {cerebro.broker.getvalue():,.2f} USDT')
```

```
Buying at 43544.03
current situation 43.115097262981116 3.5772395896638045 36.40307957051602
Buying at 42929.96
current situation 44.94279754443065 22.242242225031077 26.487278767890725
Buying at 43445.95
current situation 43.40502957117239 25.507267175999004 33.81758000231345
Buying at 42283.89
current situation 44.62607933521448 30.68428232205042 31.90605553919334
Buying at 42247.46
current situation 42.123857784937 11.40808990955702 28.910998065869435
Buying at 35771.37
current situation 44.4344907987664 44.43985989833891 27.483643065647122
Selling at 36798.0
Buying at 36777.38
current situation 39.74725432274282 14.512201137251395 31.82495834228603
Selling at 37703.93
Buying at 38171.67
current situation 44.06124397628028 53.11944719551684 31.021035328437375
Ending Portfolio Value: 40,534.76 USDT
```

```
Buying at 42247.46
current situation 42.123857784937 11.40808990955702 28.910998065869435
Buying at 35771.37
current situation 44.4344907987664 44.43985989833891 27.483643065647122
Selling at 36798.0
Take Profit Price: 37533.96
Stop Loss Price: 36062.04
Buying at 36777.38
current situation 39.74725432274282 14.512201137251395 31.82495834228603
Selling at 37703.93
Take Profit Price: 38458.01
Stop Loss Price: 36949.85
Buying at 38171.67
current situation 44.06124397628028 53.11944719551684 31.021035328437375
Ending Portfolio Value: 40,534.76 USDT
```

In [128]:
```
strategy2 = result[0]
print('Sharpe Ratio:', strategy2.analyzers.mysharpe.get_analysis()['sharperatio'])
analysis_drawdown = strategy2.analyzers.drawdown.get_analysis()
analysis_trades = strategy2.analyzers.tradeanalyzer.get_analysis()

gross_profit = analysis_trades['pnl']['gross']['total']
net_profit = analysis_trades['pnl']['net']['total']
total_closed_trades = analysis_trades['total']['closed']

print(f'Gross Profit: {gross_profit:.2f}')
print(f'Net Profit: {net_profit:.2f}')
print(f'Total Closed Trades: {total_closed_trades}')
```

```
Sharpe Ratio: 0.8227504996334666
Gross Profit: 36680.39
Net Profit: 30891.02
Total Closed Trades: 195
```

Other parameters

```
In [129]: max_drawdown = analysis_drawdown['drawdown']
          print("Max Drawdown", max_drawdown)

          Max Drawdown 3.8775885107041805
```

```
In [130]: max_duration = analysis_trades.len.max
          print("Max duration of single trade", max_duration)

          Max duration of single trade 3490
```

```
In [131]: average_winning_trade = analysis_trades.pnl['gross']['average']  # or 'net' depending on your preference
          average_losing_trade = abs(analysis_trades.pnl['net']['average'])  # Take the absolute value

          risk_reward_ratio_model = average_winning_trade / average_losing_trade if average_losing_trade > 0 else 0

          # Print the calculated Risk-Reward Ratio
          print(f'Risk-Reward Ratio of the Model: {risk_reward_ratio_model:.4f}')

          Risk-Reward Ratio of the Model: 1.1874
```

8. FUTURE SCOPE:

While we, as a team, tried our best to develop the best working model in the given 2 weeks we had for this project, we do recognize that there are future scopes to our project which would in our analysis, improve the present working model, manyfold.

a. Dynamic Feature Selection:

i. **Purpose:** Understanding the importance of features helps identify which indicators contribute the most to the predictive power of the model. This is crucial for refining the model and focusing on the most relevant signals.

ii. **Techniques:**

- Permutation Importance: Shuffle the values of a single feature and observe the impact on the model's performance. Higher degradation in performance indicates higher importance.
- Tree-based Models Feature Importance: For algorithms like Random Forests or Gradient Boosted Trees, intrinsic feature importance measures are available. These measures assess the contribution of each feature to the model's decision-making process.

iii. **Implementation:**

- After training the model, analyze feature importance scores. Visualize importance scores using bar plots or other visualization techniques.

b. End-to-End Development:

i. **Purpose:** End-to-end development envisions a comprehensive and unified approach, streamlining the entire software development lifecycle for enhanced efficiency and cohesion.

ii. **Techniques:**

- **Unified Workflow:** Implement a seamless workflow covering ideation, development, testing, and deployment.

- **Automation Integration:** Incorporate automation tools to streamline repetitive tasks and enhance productivity.
- **Collaboration Platforms:** Utilize collaborative platforms to facilitate communication and coordination among development teams.
- **Continuous Integration/Continuous Deployment (CI/CD):** Implement CI/CD pipelines for automated testing and continuous delivery.
- **Quality Assurance Strategies:** Integrate robust quality assurance strategies to ensure software reliability and functionality.

iii. **Implementation:**

- **Unified Toolchain:** Develop a unified toolchain that seamlessly integrates various development tools.
- **Automated Testing Frameworks:** Implement automated testing frameworks to ensure code quality and minimize errors.
- **Cross-Functional Teams:** Foster collaboration among cross-functional teams for a holistic development approach.
- **DevOps Practices:** Adopt DevOps practices to bridge the gap between development and operations.
- **User-Centric Design:** Prioritize user-centric design principles to enhance the end-user experience.

c. Ensemble Methods:

i. **Purpose:** Ensemble methods combine the predictions of multiple models to improve overall performance, robustness, and generalization.

ii. **Techniques:**

- **Bagging (Bootstrap Aggregating):** Build multiple models on different subsets of the training data and average their predictions. Random Forest is a common bagging ensemble using decision trees.
- **Boosting:** Sequentially build models, each correcting the errors of the previous ones. Common boosting algorithms include AdaBoost, Gradient Boosting Machines (GBM), and XGBoost.
- **Stacking:** Combine predictions from multiple models as inputs to a meta-model (or blender) that makes the final prediction.

iii. **Implementation:**

- Train individual models using different algorithms or configurations.
- Combine predictions through averaging, weighted averaging, or using a meta-model.
- Optimize the ensemble strategy based on performance metrics.

d. Q-Learning / Reinforcement Learning:

i. **Purpose:** Reinforcement learning equips algorithmic agents to learn optimal strategies by interacting with environments and receiving feedback through rewards or penalties.

ii. **Techniques:**

- **State Representation:** Define the state using indicators like MACD, RSI, ADX, SMA.
- **Action Space:** Specify actions: buy, sell, or hold positions.
- **Reward System:** Establish a feedback system reflecting successful or unsuccessful actions. Experiment with reward shaping for behaviour guidance.
- **Exploration-Exploitation:** Implement a balance between exploration and exploitation.

iii. Implementation:

- **Algorithm Choice:** Select a suitable RL algorithm such as DQN, PPO, or A2C.
- **Agent Architecture:** Design a neural network as the policy network.
- **Training Process:** Train iteratively, updating based on observed rewards and implementing continuous learning mechanisms.
- **Risk Management:** Ensure alignment with acceptable risk levels.
- **Backtesting:** Evaluate historical performance through backtesting. Also, consider transaction costs and slippage.
- **Fine-Tuning:** Continuously analyze and fine-tune based on ongoing evaluation

9. CONCLUSION: