Predicting Bitcoin Prices: A Comparative Study of LSTM, Random Forest, and SVM Approaches

Bob Hampton
Dept. of Computer Science
Stevens Institute of Technology
NJ, USA
rhampton@stevens.edu

Lorraine Nunes
Dept. of Computer Science
Stevens Institute of Technology
NJ, USA
Lnunes@stevens.edu

Adisesh Yeragudi
Dept. of Computer Science
Stevens Institute of Technology
NJ, USA
ayeragud@stevens.edu

***Abstract*—Our study attempts to compare various machine learning methods in their ability to accurately predict future prices of Bitcoin (BTC) using only historical pricing data. Our data is sourced via an Alpaca API [1] and consists of time-series pricing data dating back to January 1,2021. We are attempting to use an LSTM as they can naturally be applied to time-series data, Random Forests as they provide robustness to overfitting, and Support Vector Machines (SVMs) as they do well on high-dimensional data. Our goal is to compare and contrast the aforementioned machine learning algorithms on the basis of accuracy, overfitting, inference time, and resources used during training. Our experimental results show that LSTMs show a natural fit to predicting BTC prices over the other approaches mentioned. Our study aims to comprehensively compare a variety of machine learning approaches in order to show how they may or may not be well suited for problems of a similar class.**

I. Introduction

The trading of securities, and the prediction of their prices have been of interest since the advent of exchanges. A financial security can be described as a tradable financial instrument that represents a claim on an underlying asset. Our efforts in this study is to compare and contrast different machine learning methods in their efficacy of predicting bitcoin (BTC) prices.

Our dataset comprises of historical BTC price data collected from the Alpaca Crypto Historical Data API [1]. Alpaca provides users access to pricing data for a variety of financial markets [1]. We will primarily use historical pricing data in order to conduct technical analysis of this information as opposed to using alternative forms (news, investment amounts of firms, etc.) in order to make comparisons.

The machine learning algorithms we will use in order to conduct this study, would be a Long Term Short Memory (LSTM) neural network, a Random Forest approach, and a Support Vector Machine (SVM) approach. The selection criteria for these methods was dependent on comparing machine learning methods with significant and interesting advantages and drawbacks to each. LSTMs tend to perform well on temporal and sequential data, however may require a significant amount of data compared to SVMs and Random Forests. Random Forests, being an ensemble learning method, can prove to be more robust on and have lower variance on validation sets when compared to SVMs and LSTMs, however they can prove to be inefficient in inference comparatively. When considering SVMs, the ability to succeed on high dimensionality and smaller datasets, might prove beneficial, however the sensitivity to noisy data may prove it to be less successful than the other methods. Our goal is to compare whether a neural network that naturally fits the temporal nature of stock prices can outperform a method that is quite robust in terms of variance (Random Forests) or a method that can succeed on smaller amounts of data.

Our current baseline model, the LSTM, shows an average deviation of less than 2% from actual prices on the validation set. Considering this, it outpaces or is on par with similar strategies.

II. Related Work

There are various algorithms that can predict with a suggestion of when it is best to buy or sell stock. According to Investopedia, Algorithmic Trading places trades at specific moments faster than humanly possible [2]. It is based on volume-weighted average price or time-weighted average price. It buys 20 stocks when the 20-day moving average exceeds the 100-day moving average, and sells when the stock goes below the 100-day average [2].

Trend following algorithms focus on the momentum of stock prices, typically using 50-day and 200-day moving averages. According to Quantified Strategies, these algorithms exit trades when trends are no longer followed, with no forecasting involved [3]. Whether a trend exists can be subjective—some analysts draw lines between chart tops and bottoms [2].

Arbitrage opportunities involve buying stock at a lower price in one market and selling it at a higher price in another. Harvard Business School Online notes this profit differential is short-lived, often requiring large volumes, and is commonly used by hedge funds [4]. Arbitrage types include pure, merger, and convertible [2].

Mean reversion assumes that high and low prices eventually return to a mean value. TradingWithThePros explains this involves detecting when stock prices significantly deviate from their historical average [5]. An example indicator is the Stochastic Oscillator, which follows 20% and 80% levels [2].

Percentage of volume strategies send partial orders based on participation ratio and traded volume, self-regulating when user-defined price thresholds are reached. Quantified Strategies describes it as ideal for large-volume trades while avoiding major market disruptions [2].

III. Our Solution

Our solution involves using machine learning algorithms as a representative of various classes - LSTMs for neural network approaches, Random Forests for ensemble learning approaches, and SVMs to represent supervised learning that succeeds on high dimensionality. Our approach to finding an algorithm that is best suited for predicting future BTC prices involves comparing and contrasting results based on: model accuracy, model overfitting, training resources, and inference time. Our aim is to provide context regarding tradeoffs of each algorithm depending on the user's needed emphasis.

A. *Description of Dataset*

The dataset used in this project consists of hourly Bitcoin (BTC) price data for the BTC/USD trading pair, retrieved from the Alpaca Crypto Historical Data API [1]. For the LSTM model, the data spans from January 1, 2021 to the current date and includes key features such as open, high, low, close, volume, trade count, and VWAP (Volume Weighted Average Price) for each hourly interval.

A thorough data quality assessment confirmed the absence of missing values across all columns. Outlier detection, conducted using the Interquartile Range (IQR) method [6], revealed no anomalies in the closing price—the primary feature used in prediction—reinforcing the dataset's overall integrity and reliability. However, a substantial number of outliers were identified in the volume column, which is a critical input for our LSTM model. To determine their significance, we visualized the BTC closing price alongside volume data and the detected outliers in Figure 1.

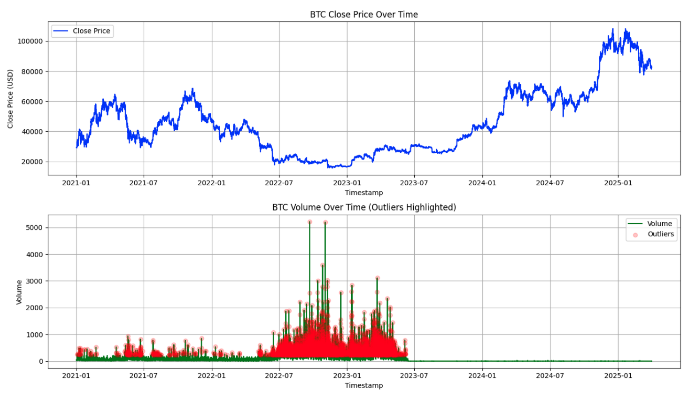

Fig. 1. Volume outliers compared to actual price.

The analysis revealed that these volume spikes aligned with periods of heightened market activity and price volatility—characteristic behavior in cryptocurrency markets. Rather than excluding this information, we opted to retain the volume outliers, recognizing their potential to enhance the model's ability to learn from sharp, unpredictable market shifts.

To enhance the forecasting capability of our LSTM model, we performed extensive feature engineering. Each hourly observation is augmented with a variety of technical indicators commonly used in algorithmic trading. These include:

- **Momentum Indicators:** Relative Strength Index (RSI), Stochastic Oscillator (%K, %D), On-Balance Volume (OBV)
- **Trend Indicators:** Moving Average Convergence Divergence (MACD), Average Directional Index (ADX), Commodity Channel Index (CCI)
- **Volatility Indicators:** Bollinger Band Width, rolling standard deviation
- **Price Action Features:** moving average ratio (close / MA20), daily price range, candlestick body size
- **Volume Normalization:** z-scored volume feature

The complete feature vector for each time step includes 16 engineered signals, capturing various dimensions of price movement, trend strength, and market volatility.

a) Handling Volatility: To focus training our LSTM model on informative periods, we filtered the dataset to retain only the top 30% most volatile windows (based on rolling standard deviation over a 20-hour window). This ensures that the model trains on dynamic market behavior, where prediction signals are most valuable, and reduced training time significantly.

b) Preprocessing Pipeline: The following steps were applied prior to LSTM model training:

- Missing values were handled by dropping rows with nulls in any of the indicator columns.
- Each 120-hour window is used to generate one training example.
- All features were standardized using `StandardScaler`, applied independently to each sequence.
- The target variable is the scaled log return of the next closing price, multiplied by 100 for numerical stability.

This preparation pipeline ensures that both feature dynamics and temporal dependencies are captured for effective LSTM training. The resulting dataset is robust, noise-filtered, and aligned with real-world trading features.

B. *Machine Learning Algorithms*

To solve the problem of predicting short-term Bitcoin (BTC) price movements, we plan to explore three machine learning algorithms:

The first algorithm we have chosen is support vector machines due to the fact that it handles non-linearity better than something like logistic or linear regression [7]. There are a lot of factors that go into the prices of stocks, and in our case can be used for Bitcoin as well. We are interested in: moving averages, relative strength index, and moving average convergence divergence (MACD) [8]. We also want to use volatility measures like Bollinger Bands and Average True Range. Volume-based features such as bid-ask spread, market depth, and whale transactions (e.g., large buy/sell walls and major Bitcoin transfers) significantly affect price shifts [9]. Additionally, we must consider blockchain-related signals like hash rate, Bitcoin address activity, and exchange flows to assess miner confidence and wallet movement trends [10].

Since Bitcoin is highly volatile, certain hyperparameters must be considered carefully, such as the kernel type and regularization to prevent overfitting. Grid search can be applied to tune the gamma value [7]. To generate predictions, we use a signal-based approach—classifying buy, hold, or sell with +1, 0, and -1 respectively [9]. Strategy returns are calculated as the product of the signal and price change, and cumulative returns represent the overall percentage change in investment over time [10].

The second algorithm we chose is Random Forests, which is a type of ensemble learning designed to combine the output of multiple decision trees to reach a single result. The use of ensemble learning and random forests improves the robustness to overfitting of the model, which we hope will prove useful in a volatile market of BTC pricing. It will also help in showing with features and feature set will be useful for future predictions.

The random forest approach will take hourly features such as open, high, low, close, volume, VWAP, etc., as input features. We will initially begin with using 100 decision trees as a baseline, and readjust accordingly based on cross-validation results. We will also set a maximum initial depth of 10, with 5 samples per leaf. In order to predict future prices, we will set the target variable as the close price at each hour interval (the price at the end of the interval). This way we can keep the time-series data within consideration. The random forest will contain CART-like trees and train via bagging (bootstrap aggregating). Our validation set will be 20% of our data while we shall save 80% for training.

The third algorithm we chose is Long Short-Term Memory (LSTM), which is a type of recurrent neural network (RNN) designed to learn from sequential data and retain long-term dependencies. This makes it highly effective for financial time series problems where future values are influenced by previous patterns. Unlike traditional feedforward networks, LSTMs can account for the order and temporal structure of price data, which is critical in modeling trends and reversals in crypto markets.

C. Implementation Details

Implementation of SVM was done over a period from 2021-01-01 to 2024-04-01, with a 20% test and 80% training set. Several technical indicators were used for this model such as macd, Bollinger h and l for highs and lows, ema 12, ema 26 (these are for exponential moving averages), rsi with overbought and oversold levels, stochastic, adx. For any NA values, they were dropped as part of pre-processing and any data missing was filled in as well. These technical indicators were used to show the closing price among these indexes.

For SVM, we made use of GridSearchCV to handle the hyperparameters: C- this sets up the boundary, gamma-influence of a single training example, kernel-rbf (good for nonlinear decisions). When it comes time to plot the cumulative strategy return, we can see in the graph that from 2021 to 2024, it is static (this stays at 1). During this period, it is not predicting any profit or losses. This can also be based on the training data. From 2024 on, we can see a drastic decline in cumulative strategy return – proving SVM is not a very good algorithm for predicting bitcoin prices. When you look at both cumulative return and buy and hold, you can see that without this algorithm just by the nature of bitcoin and its volatile nature, buying it from 2021 and holding it until 2025 does produce profits as well as some notable losses.

Looking at accuracy, precision and recall, this model was only able to accurately predict about 22% of the time, it does not ever accurately predict hold and for sell(1), it is slightly better at 50% of the time, but it is not a great improvement.

Comparing SVM with classification and regression, we can see in Figure 2 that this algorithm is not adequate for bitcoin predicting prices either:

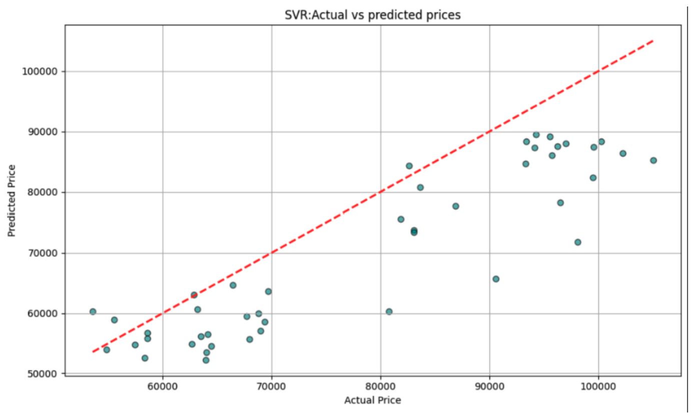

Fig. 2. SVR: Actual vs predicted prices. MAE of 9015.37 and a RMSE of 10830.19.

For both MAE, MSE and RMSE the deviation from the actual price to its predicted price is highly significant. For R2, it shows how much variance it can explain, which it does this for 54.84% of the time- which is moderate. A suggestion for this type of model would be to look into more technical indicators, handle hyperparameters better, and look into how it would handle at a shorter time range.

The second algorithm we employed is a Random Forest Regressor that was chosen due to its ensemble approach. We predicted that the ensemble based approach of the Random Forest would prove to be reliable against high levels of variance. We begin with implementing a regression model containing of 100 trees and initial depth of 10. We use 80% of our data for training and the remaining 20% for validation. Upon the creation of a base model, we use Grid search to train multiple implementations of a Random Forest Regressor by iterating through hyperparameters and comparing models. This model will then be compared to the base model in terms of performance based on MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error).

The third algorithm we employed is a Long Short-Term Memory (LSTM) network—an advanced Recurrent Neural Network (RNN) architecture designed to capture temporal dependencies in sequential data, making it ideal for financial time series such as BTC prices. Unlike feedforward networks, LSTMs retain information across time steps, enabling the model to learn patterns like trend reversals, momentum shifts, and volatility spikes.

Our implementation goes beyond basic LSTM design by integrating technical indicators, uncertainty quantification, and advanced training heuristics. The core architecture is:

- **Two LSTM layers** (each with 64 units) with dropout regularization (20%) applied after each.
- **Final dense layer** with linear activation for single-step return prediction.
- Trained using the Adam optimizer and Mean Absolute Error loss.

a) Input Sequence and Feature Set: Each training sample is constructed using a 120-hour sliding window. The model input includes 16 engineered technical features:

- **Price/Volatility:** close price, 20-hour rolling std, daily range
- **Momentum:** RSI, Stochastic Oscillator (%K, %D), OBV
- **Trend:** MACD line, signal, histogram, ADX, CCI
- **Volume & Positioning:** z-scored volume, VWAP ratio, moving average ratio

All features are scaled independently for each sample using a `StandardScaler`, and the target variable is the scaled log return of the next closing price (multiplied by 100 for numerical stability).

b) Uncertainty Modeling with Monte Carlo Dropout: To improve model interpretability and account for volatility in crypto markets, we implement Monte Carlo (MC) Dropout:

- During prediction, we perform multiple forward passes (`training=True`) to sample a predictive distribution.
- The model outputs the *mean prediction* and *standard deviation* for each forecast.
- This enables us to form 95% confidence intervals for price predictions.

c) Evaluation Protocol: We assess performance over multiple horizons:

- **EOD forecasting:** Predicting end-of-day close prices using rolling windows.
- **Metrics:** Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

IV. Comparison

We compared the predictive accuracy of the hybrid LSTM model against traditional machine learning approaches using the following metrics:

- **MAE (Mean Absolute Error)**: average deviation from true EOD price
- **RMSE (Root Mean Squared Error)**: penalizing larger errors more

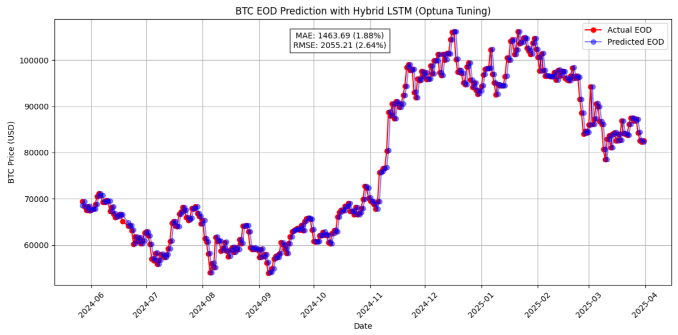

Fig. 3. BTC price prediction using an LSTM neural network trained with Optuna Tuning from January 2021 to April 2025. The model was evaluated on the final 20% of data and achieved MAE of 1463.69 (1.88%) and a RMSE of 2055.21 (2.64%). The predicted prices (blue) track actual BTC prices (red) with strong accuracy.

The baseline LSTM model demonstrated strong predictive performance, achieving a Mean Absolute Error (MAE) of 1465.34 (1.88%) and a Root Mean Squared Error (RMSE) of 2054.86 (2.64%). These metrics highlight the model's ability to capture temporal dependencies and generalize effectively to unseen data. To further optimize performance, we applied Optuna for hyperparameter tuning across 30 trials, selecting the best configuration based on validation error. With the tuned parameters, the MAE slightly improved to 1463.69 (1.88%), a marginal decrease of 1.43, while the RMSE increased insignificantly to 2055.21 (2.64%), reflecting a minor change of +0.59. Despite these small variations, the model consistently maintained high accuracy, closely tracking actual BTC price movements—as illustrated by the strong alignment between predicted (blue) and actual (red) values in Figure 3. These findings confirm the model's robustness in capturing both short-term volatility and longer-term price trends, even beyond the distribution of the training data.

A. Comparisons between models in study

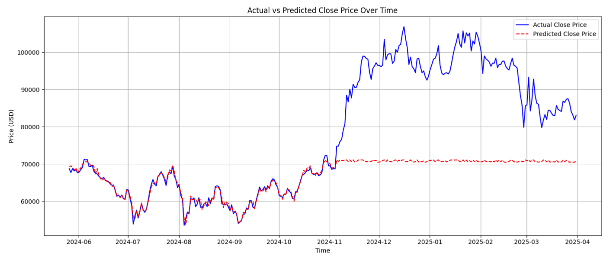

Fig. 4. BTC price prediction using a Random Forest approach trained from January 2021 to April 2025. The model was evaluated on the final 20% of the data. MAE and RMSE were 10517.525 and 15826.357

Other models used included in this study are a Random Forest approach and an SVM approach. We first look at the Random Forest approach as seen in Figure 4 above.

We observe that the Random Forest regression model was able to track the actual BTC price relatively well during periods of moderate fluctuation. However, once the price of BTC began to rise sharply beyond the range of values encountered during training, the model's predictive performance deteriorated, which can be seen in Figure 4 around November 2024. This limitation arises from two key factors. First, the nonparametric nature of Decision Trees—upon which the Random Forest is built—prevents effective extrapolation beyond the scope of the training data. Second, the model lacks an inherent understanding of temporal structure, as Random Forests treat each input independently without accounting for sequential dependencies. As a result, when faced with a sudden shift in price trends that were not reflected in the historical data, the model failed to capture the new pattern and could not accurately forecast the rapid upward movement in BTC prices. The MAE and the RMSE on the training data was appromixately 179.800 and 319.869 respectively, which shows that while there were a few missed predictions that were penalized, these are acceptable and desirable error statistics for BTC prices. On the test dataset however, our MAE and RMSE were 10517.525 and 15826.357 respectively, indicating a significant failure in prediction, and further fortifying the data seen in the graph.

In contrast with Random Forest and LSTM, both SVC (for classification) and SVR (for regression) were used. For SVC, instead of predicting price, it focused on direction (up, down, sideways within a threshold). Like Random Forest, treats each instance independently and does not know how to handle the sequence of events leading up to a sharp price change. The extrapolation issue is then seen in degradation in signal accuracy. There is a threshold sensitivity during volatility, so the SVC model is sensitive to predefined thresholds for buy, sell and hold. Especially for hold, the model was not able to accurately predict hold signals. For SVR, which focused on future price as well, but it also struggled with sharp price movements. C and epsilon during extreme events were not optimally configured to handle data points outside of training distribution. Even though lagged prices were used, it did not capture the dynamic nature of the market behavior during extreme events. When compared to the ability of the LSTM to capture temporal data and utilize a sliding window approach, the Random Forest Regressor and Support Vector Machine models are unable to maintain performance during times of high volatility outside of its existing knowledge.

B. Takeaway

Overall, the hybrid LSTM model outperformed the Random Forest and SVM models when comparing MAE and RMSE:

Model	**MAE**	**RMSE**
Hybrid LSTM (Optuna Tuned)	**1463.69**	2055.21
Hybrid LSTM (default)	1465.34	**2054.86**
SVM	9015.37	10830.19
Random Forest	10517.53	15826.36

TABLE I. MODEL COMPARISON BASED ON PREDICTIVE ACCURACY (TEST SET)

V. FUTURE DIRECTIONS

Given an additional 3–6 months, several key enhancements could be explored to further advance the effectiveness and applicability of our LSTM-based Bitcoin prediction system.

First and foremost, our goal is to transition the current predictive framework into a fully operational live trading bot. This would involve integrating our LSTM model with real-time data feeds and execution systems, such as the Alpaca trading API. We plan to initially deploy the model in a simulated environment using a paper trading account to rigorously evaluate live performance under realistic market conditions. Once the strategy demonstrates consistent profitability and risk management, we would migrate to a real-money deployment with dynamic position sizing and capital allocation.

Additionally, we aim to explore multi-model ensemble strategies by incorporating Random Forest and Support Vector Machine (SVM) models alongside our LSTM. Rather than simply comparing their performance, these models could provide secondary signals to enhance trade confidence. For instance, we could design a voting or weighting mechanism where trades are only executed when both the LSTM and at least one of the auxiliary models agree on the predicted market direction. Alternatively, the SVM and Random Forest models could be used to filter out uncertain trades or adjust trading thresholds during volatile periods.

For Random Forests, we demonstrated that within a finite and previously observed range of values, the model was able to produce accurate predictions. However, its performance deteriorated when faced with values or patterns outside of the training distribution. To address these limitations and enhance predictive performance in dynamic time series environments, further strategies can include more advanced feature engineering—such as incorporating lag variables, rolling statistical metrics, and time-based indicators—as well as exploring hybrid modeling approaches. Integrating Random Forests with models that are inherently time-aware, such as recurrent neural networks (RNNs), LSTMs, or Transformer-based architectures, may also improve the model's ability to capture temporal dependencies and adapt to rapidly changing trends.

VI. CONCLUSION

In conclusion, all our models were observed over daily data from 2021-1-1 to 2025-04-01. For preprocessing, we fill in and drop any NA data, splitting our train and test sets as 80% and 20% respectively. Key features added include open, high, low, close, volume, trade count and VWAP. Technical indicators were part of our feature engineering to make use of moving averages, volatility measures, and in the case of LSTM – bid-ask spread, market depth and whale transactions for volume-based features were also added.

When comparing LSTM, Random Tree Forest and Support Vector Machines, LSTM is far superior to the other two when it comes to predicting bitcoin prices. It only deviates less than 2% from actual prices and it does very well in generalizing "unseen data". LSTM performs well on temporal and sequential data, as highlighted previously on this paper, however it needs significantly more data than SVM and Random Forest. Random Forest proves to have a lower variance compared to the other models. When it comes to SVM, the sensitivity to noisy data makes it far less ideal than the other two models.

REFERENCES

[1] A. Markets, "Crypto historical data api documentation," 2025. [Online]. Available: https://docs.alpaca.markets/docs/historical-api

[2] Investopedia, "Basics of algorithmic trading: Concepts and examples," 2025, accessed March 2025. [Online]. Available: https://www.investopedia.com/terms/a/arbitrage.asp

[3] Quantified Strategies, "Trend following trading strategies and systems (backtest results)," 2025, accessed March 2025. [Online]. Available: https://www.quantifiedstrategies.com/trend-following-trading-strategy

[4] Harvard Business School Online, "What is arbitrage? 3 strategies to know," 2025, accessed March 30, 2025. [Online]. Available: https://online.hbs.edu/blog/post/what-is-arbitrage

[5] Trade with the Pros, "Mean reversion strategies: A guide to profitable trading," 2025, accessed March 30, 2025. [Online]. Available: https://tradewiththepros.com/mean-reversion-strategies

[6] J. W. Tukey, *Exploratory Data Analysis*. Addison-Wesley, 1977.

[7] A. Ng, "Machine learning specialization," Coursera, Stanford University, 2020, provides foundational knowledge on Support Vector Machines (SVM) and hyperparameter tuning.

[8] J. J. Murphy, *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance, 1999.

[9] B. Academy, "A beginner's guide to machine learning in crypto trading," 2023. [Online]. Available: https://academy.binance.com/en/articles/a-beginners-guide-to-machine-learning-in-crypto-trading

[10] QuantInsti, "Support vector machines (svm) in algorithmic trading," 2023. [Online]. Available: https://blog.quantinsti.com/support-vector-machine/