Predicting Bitcoin Prices Using Long Short-Term Memory (LSTM) in Time Series

Dwi Budi Santoso[1,a)], Arief Jananto[1,b)], Dewi Handayani, U.N.[1,c)] , Jeffri Alfa Razaq[1,d)]

[1,2,3,4]*Faculty of Information Technology and Industry, Universitas Stikubank, Semarang, Indonesia*

[a)]Corresponding author: dbs@edu.unisbank.ac.id

[b)]ajananto@edu.unisbank.ac.id

[c)]dewi_h@edu.unisbank.ac.id

[d)]mrjf@edu.unisbank.ac.id

Abstract. This study aim to predict Bitcoin prices using Long Short-Term Memory (LSTM) in a time series analysis. The dataset used in this research is obtained from Yahoo Finance, covering the period from January 1, 2017, to December 31, 2022. The LSTM model is trained on this dataset to capture the patterns and trends in Bitcoin price movements. The data undergoes preprocessing, including feature selection where only the date and closing price columns are retained. The closing price column is transformed into the "Price" variable. The dataset is then normalized using the Min-Max scaler. It is split into training and testing sets to evaluate the model's performance. The LSTM model architecture comprises multiple layers with dropout regularization to prevent overfitting. The model is trained on the training set and evaluated on the testing set. The experimental results demonstrate that the LSTM model successfully predicts Bitcoin prices with low loss and mean absolute percentage error (MAPE). The achieved test loss of 0.00492361793294549 indicates the model's ability to accurately predict Bitcoin prices. The MAPE of 0.11878698834238316 highlights the model's low average percentage error in predicting Bitcoin price changes. These results indicate that the LSTM model effectively captures the complex temporal dependencies in Bitcoin's time series data, resulting in accurate price predictions.

Keywords: Bitcoin, Long Short-Term Memory (LSTM), Time Series Analysis, Price Prediction.

INTRODUCTION

Bitcoin is a decentralized digital currency that operates on a peer-to-peer network[1]. It has gained significant attention and popularity due to its unique characteristics and potential to disrupt traditional financial systems. Bitcoin allows for secure and transparent transactions without the need for intermediaries such as banks or governments[2]. Its limited supply and decentralized nature make it attractive to individuals seeking an alternative store of value and a hedge against inflation. Despite its volatility and regulatory challenges, Bitcoin has emerged as a prominent player in the financial market, sparking debates about its long-term viability and potential impact on the future of money and financial transactions[3].

One of the main characteristics of Bitcoin is its highly volatile price. The price of Bitcoin has experienced extreme fluctuations from time to time, increasing investor interest. The ability to accurately predict the price of Bitcoin has become a very interesting and important topic for traders, analysts and researchers.

With advancements in data analysis and artificial intelligence, several traditional methods for time series data prediction have been developed and improved. One example is Autoregressive Integrated Moving Average (ARIMA), which models the relationship between the current observation and previous observations while addressing changes in trends and stationary properties in the data[4]. Exponential Smoothing (ES) is also a popular choice that takes into account trend changes and seasonal patterns in the data[5]. Seasonal Autoregressive Integrated Moving Average (SARIMA) is an extension of ARIMA that additionally considers seasonal components in the data[6].

However, these traditional methods have their limitations. ARIMA assumes that the data is stationary, which may not always hold true for real-world datasets[7]. ES may struggle with highly volatile or irregular data patterns[8]. SARIMA requires manual identification and tuning of seasonal components, which can be challenging and time-consuming[9]. Therefore, while these traditional methods have their merits, they may not always provide accurate predictions for all types of time series data and may require further refinement or augmentation with more advanced techniques.

Long Short Term Memory (LSTM) is a type of recurrent neural network (RNN), has emerged as a powerful approach for predicting time series data. LSTM overcomes some of the limitations of traditional methods by effectively capturing dependencies and long-term patterns in sequential data. It overcomes the challenges of modeling complex nonlinear relationships, capturing nonlinear trends, and dealing with long-term dependencies[10].

The LSTM's ability to store and utilize information over a long period of time is due to its memory cell structure, enabling it to capture short-term and long-term patterns in time series data[11]. This is especially useful for capturing trends, seasonality, and other complex patterns that may not be easy to capture with traditional methods. Additionally, LSTMs can handle variable length sequences, making them adaptable to different lengths of time series[12].

METHODS

PROPOSED METHOD

This study aims to demonstrate the potential of LSTM in predicting Bitcoin prices based on historical data analysis. The research architecture is depicted in **FIGURE** 1, illustrating the sequential steps involved in the study.

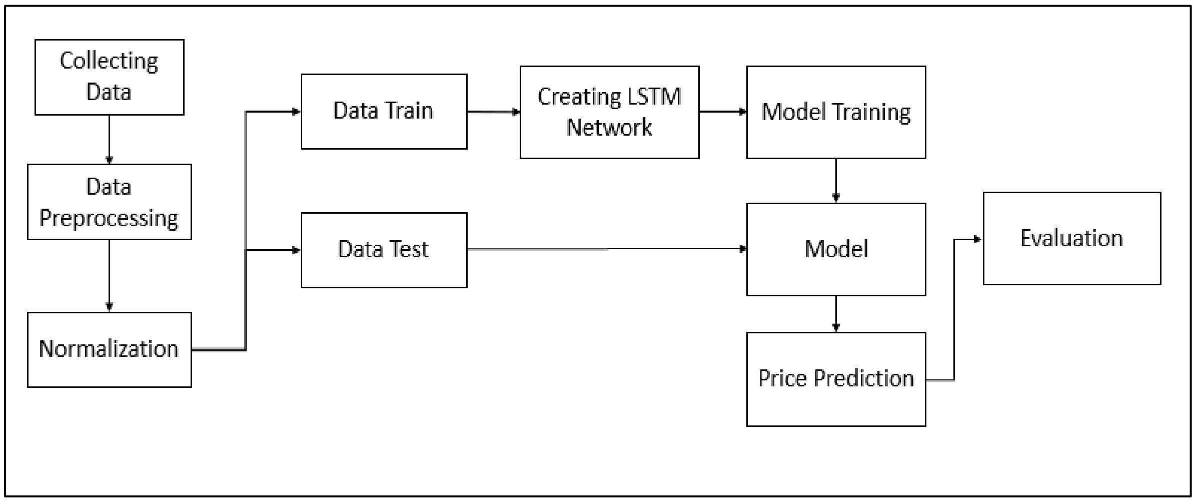

FIGURE 1. Bitcoin price prediction model

DATA SOURCE

The data source for this study is the daily Bitcoin price obtained from Yahoo Finance, spanning from 1 January 2017 to 31 December 2022. **TABLE 1** presents an example of Bitcoin price data during the specified time range.

TABLE 1. Bitcoin Prices Example

Date	**Open**	**High**	**Low**	**Close**	**Volume**
2017-01-01	963.658020	1003.080017	958.698975	998.325012	147775008
2017-01-02	998.617004	1031.390015	996.702026	1021.750000	222184992
2017-01-03	1021.599976	1044.079956	1021.599976	1043.839966	185168000
2017-01-04	1044.400024	1159.420044	1044.400024	1154.729980	344945984
2017-01-05	1156.729980	1191.099976	910.416992	1013.380005	510199008

A.DATA PREPROCESSING

During this preprocessing stage, feature selection is conducted. This process choose a subset of relevant features or variables from the available dataset to train the LSTM[13]. Feature selection aims to identify the most informative and influential features that contribute to accurate predictions while eliminating irrelevant or redundant features[14]. retaining only the "Date" and "Close" columns. The "Close" column is then transformed and renamed as "Price," as shown in **TABLE** 2.

TABLE 2. Bitcoin Price after transformed

Date	**Price**
2017-01-01	998.325012
2017-01-02	1021.750000
2017-01-03	1043.839966
2017-01-04	1154.729980
2017-01-05	1013.380005

B. NORMALIZATION AND DATA SPLITTING

Normalization is an essential step in data preprocessing to ensure that features are on a consistent scale and comparable to each other[15]. In this study, the Min-Max Scaler technique is applied for normalization. The Min-Max scaler normalization is a data preprocessing technique used to transform the input data into a specific range[16]. The purpose of applying Min-Max scaler to Bitcoin price data is to normalize the values between a predetermined minimum and maximum range, typically 0 to 1 as shown in **TABLE 3**.

TABLE 3. Bitcoin Price after normalized

Original Price	Normalized Price
998.325012	0.00330246
1021.750000	0.00365319
1043.839966	0.00398393
1154.729980	0.00564423
1013.380005	0.00352787

By normalizing the data, the Min-Max scaler ensures that all the input features are on a similar scale, which can help improve the training process of the LSTM model. This is important because LSTM networks are sensitive to the scale of the input data. If the input features have significantly different scales, it can lead to slower convergence during training and make it difficult for the model to learn meaningful patterns in the data.

The Min-Max scaler operates by subtracting the minimum value of the data and then dividing it by the range of the data (maximum value minus minimum value). This process linearly scales the data to the desired range, preserving the relative relationships between the values.

Since it is not possible to train future data in time series data, it is not permitted to split the time series data randomly. The test set is always newer than the training set. In this study, 2022 was used as test data and the others for training as sown in **FIGURE 2**.

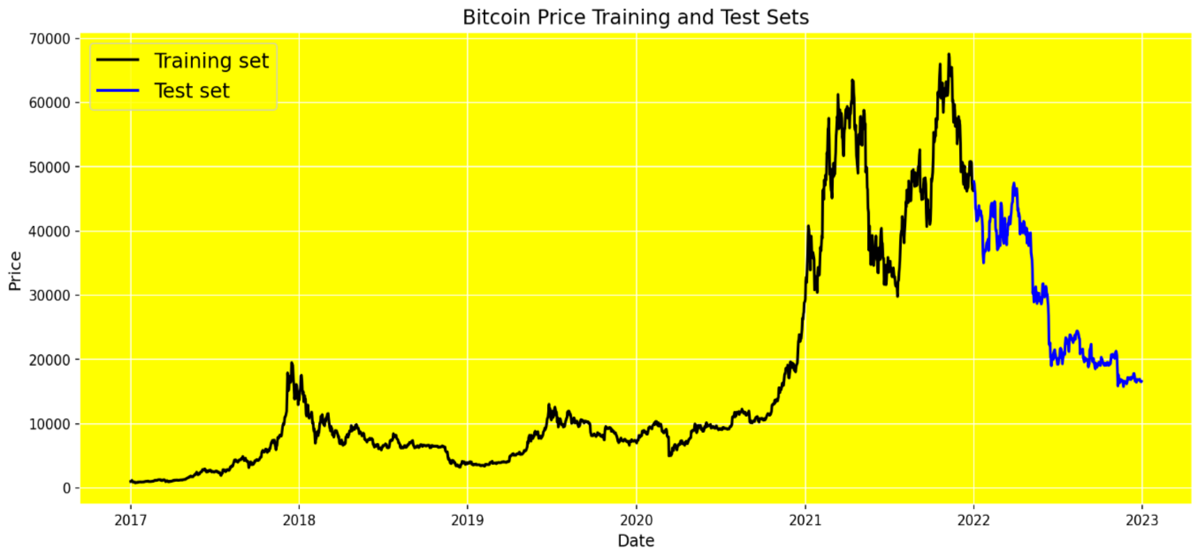

FIGURE 2. Bitcoin Price Training and Test Sets

C. CREATING SLIDING WINDOW

The implementation of creating a sliding window for Bitcoin price prediction involves several steps. Firstly, the window size is defined, which determines the number of previous time steps used as input features. In this case, the window size is set to 60, indicating that the previous 60 days' Bitcoin prices will be used to predict the next day's price.

Next, the window is initialized by selecting the initial set of data points from the beginning of the Bitcoin price dataset, spanning 60 days. Within each window, the input features are extracted, consisting of the Bitcoin prices for the previous 60 days, along with the corresponding target variable, which is the price of Bitcoin for the next day.

These extracted features and target variables are then stored, and the window is shifted by one day to the right. The feature extraction and storage process is repeated for each new window position, sliding through the dataset until the entire dataset has been covered.

By implementing this sliding window approach with a window size of 60, the Bitcoin price dataset is transformed into a supervised learning format[17]. Each window represents a training or testing sample, where the previous 60 days' Bitcoin prices act as input features, and the price of Bitcoin for the next day serves as the target variable to be predicted. The result is presented in **TABLE** 4.

TABLE 4. Data Split and Shape for Bitcoin Price Prediction

Data Split	**X Shape**	**Y Shape**
Training	(1766, 60, 1)	(1766, 1)
Testing	(364, 60, 1)	(364, 1)

This sliding window implementation allows machine learning models, such as LSTM, to be trained using the extracted features and target variables. The models can learn from the historical patterns and relationships within the Bitcoin price data and make predictions based on the available historical information.

D. LSTM MODEL TRAINING

LSTM method can be seen in **FIGURE 2**

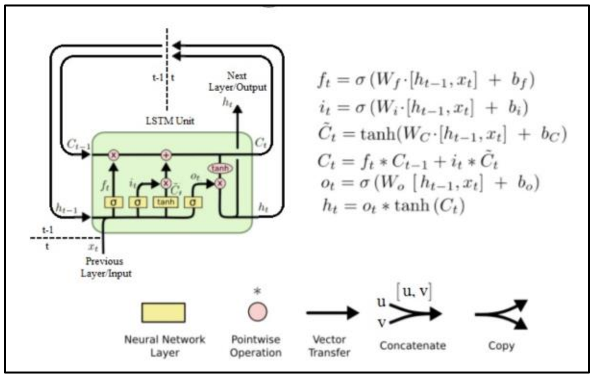

FIGURE 3. LSTM model

The theory behind LSTM is very complicated as shown in figure 2. However, a simple example that is easy to understand will be used. Imagine the following sentence.

"In Indonesia, one of the most popular rice preparations is nasi..."

Most people will probably answer fried rice. Because there are clues to the word Indonesian and popular. It can be seen that the context of a word is influenced by the words that precede it. Neural networks can do this using

LSTMs. Long Short-Term Memory (LSTM) is a technique commonly used in natural language processing that allows a model to understand the meaning of a sentence based on word order, as in the example of fried rice.

TABLE 5 provide LSTM Model Architecture used in this research.

TABLE 5. LSTM Model Architecture for Bitcoin Price Prediction

Layer Type	**Output Shape**	**Number of Parameters**	**Activation Function**
Input	(window_size, 1)	0	-
LSTM	(window_size, 64)	16640	-
Dropout	(window_size, 64)	0	-
LSTM	(window_size, 64)	33024	-
Dropout	(window_size, 64)	0	-
LSTM	(64,)	33024	-
Dropout	(64,)	0	-
Dense	(32,)	2080	Softmax
Dense	(1,)	33	-

Explanation:
1. Input Layer: It takes the input shape of (window_size, 1), which corresponds to the window size and the number of features (in this case, only one feature representing the Bitcoin price).
2. LSTM Layer 1: It has an output shape of (window_size, 64), indicating that it returns sequences for each time step and has 64 units/neurons. The number of parameters in this layer is 16,640.
3. Dropout Layer 1: It randomly drops 20% of the input units, helping to prevent overfitting by introducing regularization.
4. LSTM Layer 2: It has the same output shape as the previous LSTM layer, (window_size, 64), and 33,024 parameters.
5. Dropout Layer 2: Similar to Dropout Layer 1, it randomly drops 20% of the input units.
6. LSTM Layer 3: It returns a single output vector of size 64, representing the final hidden state of the LSTM layer, with 33,024 parameters.
7. Dropout Layer 3: It randomly drops 20% of the input units.
8. Dense Layer: It has an output shape of (32,) and applies the softmax activation function. It introduces non-linearity to the model and has 2,080 parameters.
9. Dense Layer (Output Layer): It produces a single output value, representing the predicted Bitcoin price, with 33 parameters.

The model is compiled using the mean squared error loss function and the Nadam optimizer. The summary provides a concise overview of the model's architecture, including the output shapes and the number of trainable parameters in each layer.

RESULTS AND DISCUSSION

A. MODEL RESULT

During the training of the model, several parameters are specified to control the learning process. In this case, the model is trained for 150 epochs, which means it goes through the entire training dataset 150 times. Each epoch allows the model to update its internal parameters based on the patterns it observes in the data, gradually improving its predictive ability. Additionally, a batch size of 32 is chosen, indicating that the model updates its parameters after processing 32 training examples at a time. This helps in optimizing the training process by utilizing mini-batches of data instead of processing the entire dataset at once. Furthermore, a validation split of 0.1 is used, which means that 10% of the training data is reserved as a validation set. This subset of data is used to evaluate the model's performance during training and can help in monitoring for overfitting or generalization issues.

After training the model for 150 epochs, the obtained loss value is 2.2062e-04, while the validation loss is 0.0069. The loss value represents the discrepancy between the predicted values and the actual values of the target variable during the training process. A lower loss value indicates that the model is able to make more accurate predictions on the training data. Figur 4, shown the result

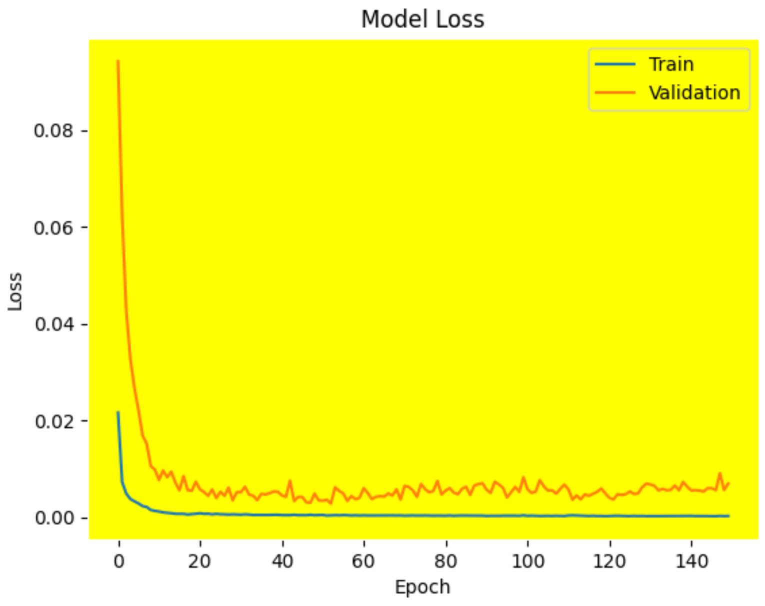

FIGURE 4. LSTM Model Result

B. EVALUATION

By evaluating the model's performance on the test dataset and comparing the predicted values (y_pred) with the actual values (y_test), it provides insights into the model's accuracy and its ability to make accurate predictions on unseen data. The MAPE metric specifically measures the average percentage difference between the predicted and actual values, providing an indication of the model's relative performance.

These evaluation metrics provide insights into the performance of the LSTM model on the test dataset. A low test loss, a low MAPE, and a high test accuracy indicate that the model has learned patterns and relationships in the data and can make accurate predictions on unseen data. These results suggest that the model performs well in predicting Bitcoin prices based on the historical data used in the study.

TABLE 6. Performance Metric

Metric	**Value**
Loss	0.00492361793294549
MAPE	0.11878698834238316
Accuracy	0.8812130116576169

As observed, the predictions generated by the LSTM model show alignment with the actual price. In addition, the Loss and Accuracy (1-MAPE) values obtained on the test data set confirms the model's excellent performance. Visualization shown in **FIGURE 5**.

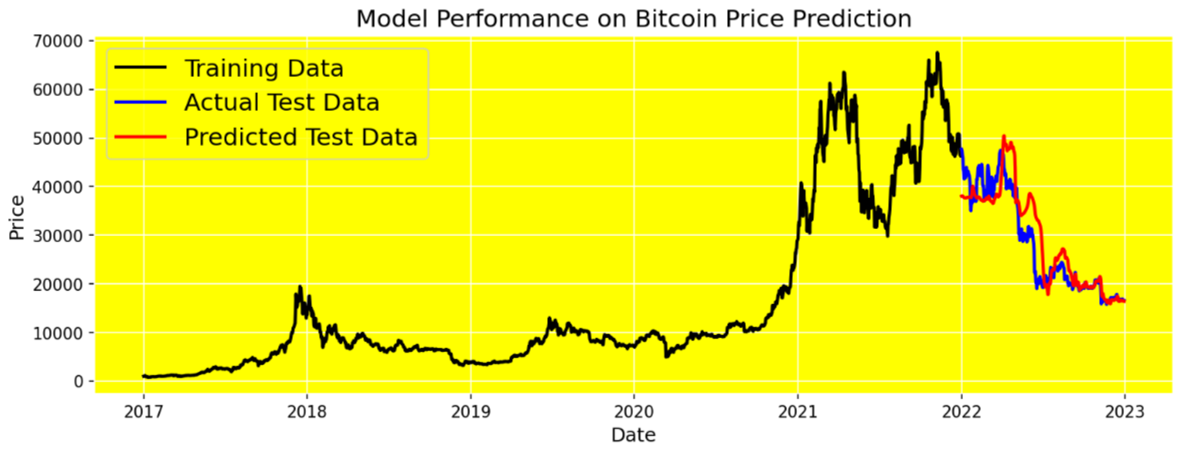

FIGURE 5. LSTM Model Performance

CONCLUSIONS

Based on the obtained results, it can be concluded that the LSTM model used performed well in predicting Bitcoin prices. With a loss of 0.00492361793294549, the model was able to generate predictions that closely match the actual prices with a high level of accuracy. Furthermore, the MAPE value 0.11878698834238316 indicates a low level of error in predicting the percentage change in Bitcoin prices. The model also demonstrated good performance in predicting the direction of Bitcoin price movements. However, it's important to note that these results may vary depending on the dataset, features, and hyperparameters used in the model training process.

The next study will explore the use of more complex LSTM architectures, such as models with multiple layers, bidirectional LSTM, or models with attention mechanisms. These advanced LSTM models can help uncover more intricate patterns and relationships in Bitcoin's time series data. By incorporating additional layers or attention mechanisms, the model can focus on relevant information and capture long-term dependencies more effectively. This exploration can provide deeper insights into the dynamics of Bitcoin's price movements and potentially enhance the accuracy of the predictions.

REFERENCES

[1] R. Sujatha, V. Mareeswari, J. M. Chatterjee, A. A. A. Mousa, and A. E. Hassanien, “A Bayesian Regularized Neural Network for Analyzing Bitcoin Trends,” *IEEE Access*, vol. 9, pp. 37989–38000, 2021, doi: 10.1109/ACCESS.2021.3063243.

[2] S. Ahamad, P. Gupta, P. Bikash Acharjee, K. Padma Kiran, Z. Khan, and M. Faez Hasan, “The role of block chain technology and Internet of Things (IoT) to protect financial transactions in crypto currency market,” *Mater Today Proc*, vol. 56, pp. 2070–2074, 2022, doi: 10.1016/j.matpr.2021.11.405.

[3] M. A. FAUZI, N. PAIMAN, and Z. OTHMAN, “Bitcoin and Cryptocurrency: Challenges, Opportunities and Future Works,” *The Journal of Asian Finance, Economics and Business*, vol. 7, no. 8, pp. 695–704, Aug. 2020, doi: 10.13106/jafeb.2020.vol7.no8.695.

[4] Y. Lai and D. A. Dzombak, “Use of the Autoregressive Integrated Moving Average (ARIMA) Model to Forecast Near-Term Regional Temperature and Precipitation,” *Weather Forecast*, vol. 35, no. 3, pp. 959–976, Jun. 2020, doi: 10.1175/WAF-D-19-0158.1.

[5] W. Sulandari and P. C. Rodrigues, “Exponential Smoothing on Modeling and Forecasting Multiple Seasonal Time Series: An Overview,” *Fluctuation and Noise Letters*, vol. 20, no. 04, p. 2130003, Aug. 2021, doi: 10.1142/S0219477521300032.

[6] I. Tyass, A. Bellat, A. Raihani, K. Mansouri, and T. Khalili, “Wind Speed Prediction Based on Seasonal ARIMA model,” *E3S Web of Conferences*, vol. 336, p. 00034, Jan. 2022, doi: 10.1051/e3sconf/202233600034.

[7] V. Kozitsin, I. Katser, and D. Lakontsev, “Online Forecasting and Anomaly Detection Based on the ARIMA Model,” *Applied Sciences*, vol. 11, no. 7, p. 3194, Apr. 2021, doi: 10.3390/app11073194.

[8] Wei, Wang, Ni, and Tang, "Research and Application of a Novel Hybrid Model Based on a Deep Neural Network Combined with Fuzzy Time Series for Energy Forecasting," *Energies (Basel)*, vol. 12, no. 18, p. 3588, Sep. 2019, doi: 10.3390/en12183588.

[9] S. Bouktif, A. Fiaz, A. Ouni, and M. A. Serhani, "Multi-Sequence LSTM-RNN Deep Learning and Metaheuristics for Electric Load Forecasting," *Energies (Basel)*, vol. 13, no. 2, p. 391, Jan. 2020, doi: 10.3390/en13020391.

[10] Y. Tian and L. Pan, "Predicting Short-Term Traffic Flow by Long Short-Term Memory Recurrent Neural Network," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, IEEE, Dec. 2015, pp. 153–158. doi: 10.1109/SmartCity.2015.63.

[11] M. Pacella and G. Papadia, "Evaluation of deep learning with long short-term memory networks for time series forecasting in supply chain management," *Procedia CIRP*, vol. 99, pp. 604–609, 2021, doi: 10.1016/j.procir.2021.03.081.

[12] Z. Shen, Y. Zhang, J. Lu, J. Xu, and G. Xiao, "A novel time series forecasting model with deep learning," *Neurocomputing*, vol. 396, pp. 302–313, Jul. 2020, doi: 10.1016/j.neucom.2018.12.084.

[13] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A Comprehensive Review of Dimensionality Reduction Techniques for Feature Selection and Feature Extraction," *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56–70, May 2020, doi: 10.38094/jastt1224.

[14] A. Suruliandi, G. Mariammal, and S. P. Raja, "Crop prediction based on soil and environmental characteristics using feature selection techniques," *Math Comput Model Dyn Syst*, vol. 27, no. 1, pp. 117–140, Jan. 2021, doi: 10.1080/13873954.2021.1882505.

[15] A. Aytekin, "Comparative Analysis of the Normalization Techniques in the Context of MCDM Problems," *Decision Making: Applications in Management and Engineering*, vol. 4, no. 2, pp. 1–25, Mar. 2021, doi: 10.31181/dmame210402001a.

[16] V. N. G. Raju, K. P. Lakshmi, V. M. Jain, A. Kalidindi, and V. Padma, "Study the Influence of Normalization/Transformation process on the Accuracy of Supervised Classification," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, IEEE, Aug. 2020, pp. 729–735. doi: 10.1109/ICSSIT48917.2020.9214160.

[17] M. Löning, A. Bagnall, S. Ganesh, V. Kazakov, J. Lines, and F. J. Király, "sktime: A Unified Interface for Machine Learning with Time Series," Sep. 2019.