Evaluating LSTM Performance on Multivariate Time Series with One-Class SVM Outlier Detection

Ragita Anillya Putri Prasetianto[1*], Sugiyarto Surono[2], Aris Thobirin[3]
[1,2,3] Department of Mathematics, University of Ahmad Dahlan Yogyakarta, Indonesia
*corr-author: putrianillya23@gmail.com

Abstract - Weekly sales forecasting plays a crucial role in retail business planning and inventory management.This study evaluates the prediction performance of a Long Short-Term Memory (LSTM) model for weekly sales forecasting after data preprocessing using standardization and outlier detection with One-Class Support Vector Machine (OCSVM) method. The independent variables used include temperature, fuel price, holidays, Consumer Price Index (CPI), and unemployment rate, with weekly sales as the target variable. The dataset is preprocessed using StandardScaler and OCSVM to detect and remove outliers before model training. The evaluation shows that the LSTM model on the clean data achieves an MSE of 0.03, an RMSE of 0.18, and an MAE of 0.11. The LSTM model demonstrates good forecasting performance when trained on cleaned data without outliers. This study provides practical insights into applying data preprocessing with OCSVM to improve the consistency of prediction models in retail time series analysis.

Keywords: OCSVM; LSTM; multivariate time series; weekly sales forecasting.

I. INTRODUCTION

Time series data is a collection of data collected consistently at specific time intervals. This data type is widely used in various fields, including economics, health, environment, and retail, to model trends, seasonal patterns, and irregular fluctuations [1,2]. However, raw data often contains outliers that complicate analysis and reduce model accuracy. Outliers are data points that appear significantly different from the rest [3].

In multivariate time series data, outlier detection becomes more complex as it must consider the relationships between variables and the temporal aspects of each variable, unlike univariate methods that only focus on one variable. Understanding these relationships allows for more accurate predictions, better trend analysis, and better detection accuracy [4,5]. This challenge highlights the importance of methods that can handle high-dimensional and temporal patterns in data.

A commonly used method for handling outliers in unsupervised data is the One-Class Support Vector Machine (OCSVM) [6]. This method is designed to distinguish between normal (inlier) and outlier data by utilizing kernels to capture non-linear patterns [7,8]. Radial Basis Function (RBF) kernels are often used because they can handle complex and non-linearly separable data distributions [9,10].

Long Short-Term Memory (LSTM) is an artificial neural network that has proven effective in analyzing sequential data such as time series [11]. According to [12], this model can accurately predict stock prices due to its ability to capture long-term patterns and temporal relationships. In addition, LSTM has also been used in dynamic sales price analysis, such as in the temporal price prediction of NFT (Non-Fungible Tokens) sales [13]. LSTM's advantages in recognizing recurring patterns and adapting to data changes over time make it the right choice for understanding complex data dynamics, including retail sales data.

However, although LSTM has shown effectiveness in predicting time series data, the quality of the input data remains a crucial factor. Therefore, preprocessing steps, such as outlier detection with OCSVM, are applied to produce cleaner data for training the forecasting model.

Previous studies, such as those conducted by [14], have discussed various time-series data preprocessing techniques, including normalization, missing value imputation, and outlier detection. However, these discussions remain general and do not explicitly examine the application of the OCSVM outlier detection method for multivariate time-series data. In addition, the datasets used in those studies, such as the AirQuality UCI dataset, have different characteristics compared to retail sales data. While [14] provided a comprehensive survey of time-series preprocessing methods, including OCSVM, their empirical evaluation did not cover retail datasets nor quantify its effect on LSTM forecasting accuracy. This study addresses that gap by applying OCSVM-based outlier detection on multivariate sales data and presenting the forecasting performance metrics of an LSTM model trained with the cleaned dataset. The evaluation uses standard error metrics, namely Mean Squared Error (MSE), Root Mean Squared Error

(RMSE), and Mean Absolute Error (MAE), to demonstrate the forecasting capability of the LSTM model after outlier removal and provide insights for practical implementation in retail time-series analysis.

II. METHOD

The research flow includes data analysis, preprocessing, predictive model implementation, and testing. The preprocessing stage includes data examination, standardization, and outlier detection using OCSVM. To predict weekly sales, the implementation model includes training and testing the LSTM model on clean and raw data. The following Fig. 1 illustrates the detailed steps of the research workflow, from data collection to predictive model evaluation.

A. Data Acquisition

The dataset used in this study is Walmart's weekly sales data obtained from Kaggle in CSV format. It is multivariate time series data sorted by the date column (Date) to maintain its integrity. This dataset was chosen because it consists of various variables (multivariate) and has a time pattern suitable for sales pattern analysis, as detailed in Table I, which presents the structure of the dataset.

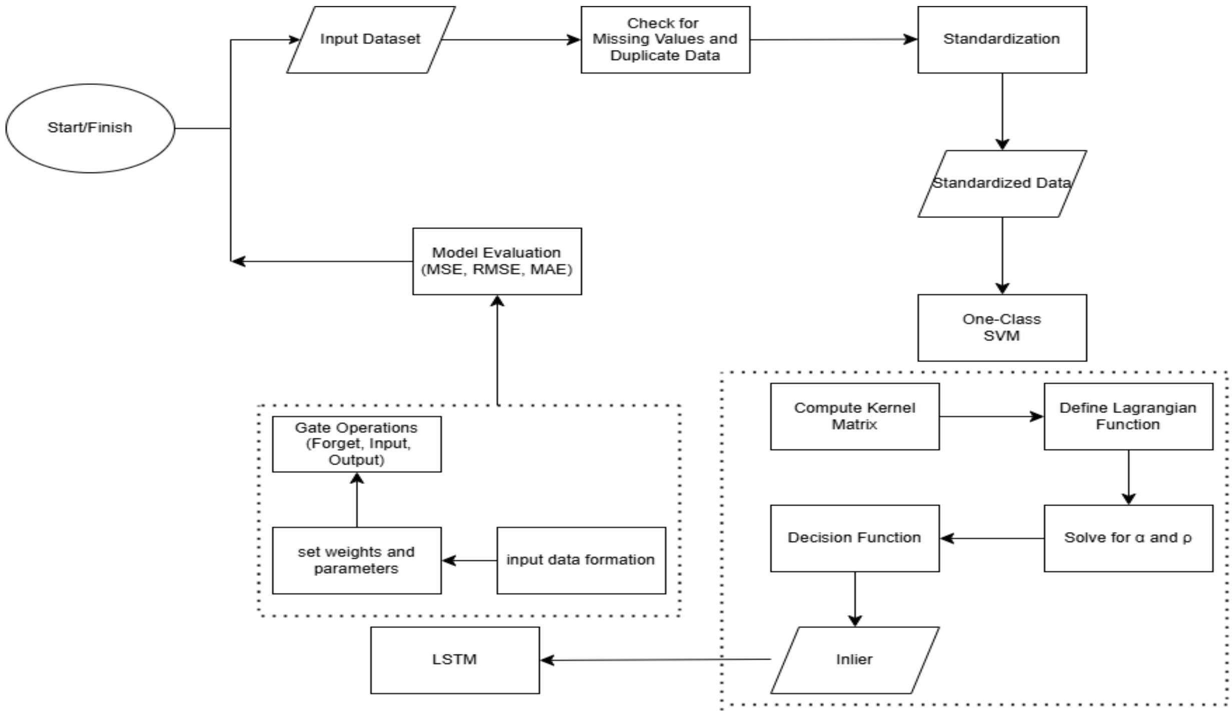

Fig. 1 Research workflow diagram

TABLE I
DATA STRUCTURE OF WALMART SALES

Variable	Frequency
Weekly Sales	Total of Weekly Sales (Target Variables)
Holiday Flag	Holiday Indicator (1: holiday, 0: not)
Temperature	Average of Temeprature (Fahrenheit)
Unemployment	Unemployment rate
CPI	Consumer Price Index

B. Data Preprocessing

Preprocessing is done to ensure the quality of the data before it is used in training the predictive model. The steps taken include:

1) Data Cleaning: at this stage, the data is checked to ensure there are no inconsistent data or missing values [15]. The results of the inspection show that the dataset has no missing or duplicate values, making it possible to proceed to the next stage.

2) Data Standardization: In the standardization process, each value in the dataset is transformed to have a mean of 0 and a standard deviation of 1. This is achieved by computing the z-score, where each value is subtracted by the mean and then divided by the standard deviation [14], as shown in (1).

$$z = \frac{x-\mu}{\sigma} \tag{1}$$

Where z = standardized value (*z-score*), x = original value, μ = dataset mean, σ = dataset standard deviation. This transformation converts each value into a measure of how far it deviates from the mean in terms of standard deviations. Standardization equalizes the scale across variables, ensuring that the model is not influenced by differences in scale and can learn all features more effectively [16]. Therefore, it is applied to all numerical features.

C. *Outlier Detection using OCSVM* [17]

One-Class Support Vector Machine (OCSVM) is a machine learning method used to detect anomalies by constructing a separating hyperplane that distinguishes normal data from outliers. This method works by mapping data to a high-dimensional feature space using a kernel function and determining a hyperplane that maximizes the margin between data points and the origin in that feature space [17].

In this approach, the main parameter is ν, which controls the maximum number of data considered outliers and the minimum number of support vectors used in the model [17].

1) Hypersphere OCSVM: Hypersphere OCSVM aims to find the smallest hypersphere that encloses the majority of normal data points in the high-dimensional feature space. Data outside this hypersphere are classified as anomalies. To account for the natural variability of normal data, OCSVM introduces slack variables (ξi). These variables allow some data points to reside just outside the defined hypersphere, tolerating minor outliers or normal variations without significantly distorting the boundary. The objective function for Hypersphere OCSVM is defined in (2) [17].

Minimize R, ξ, b

$$R^2 + \frac{1}{vn}\sum_{i=1}^{N} \xi_i \tag{2}$$

Subject to

$$\|x_i - b\|^2 \leq R^2 + \xi_i$$

$$\xi_i \geq 0, \forall i = 1, \ldots, N$$

where R is the radius of the hypersphere, and a is the center of the hypersphere.

This method works by forming a spherical decision region around the standard data, and any points outside the hypersphere are considered anomalies.

2) Objective Lagrange Function: To solve the above optimization, The Lagrangian function is constructed by introducing Lagrange multipliers. This converts the constrained problem into a solvable mathematical formulation, as shown in (3) [17].

$$L(R, b, \xi, \alpha, \beta) = R^2 + \frac{1}{vN}\sum_{i=1}^{N} \xi_i + \sum_{i=1}^{N} \alpha_i \left(\|x_i - b\|^2 - R^2 - \xi_i\right) + \sum_{i=1}^{N} \beta_i \xi_i \tag{3}$$

Where $\alpha_i \geq 0$ are Lagrange multipliers that determine the contribution of each support vector in the separation of the data, and $\beta_i \geq 0$ are additional multipliers to lower bound the slack variables ξ_i.

The optimal solution can be found by finding the partial derivatives concerning R, b, ξ, which results in a system of equations that can be solved numerically.

3) Kernel: Kernels map the data to a high-dimensional feature space to allow for better separation between normal and anomalous data. One of the most commonly used kernels in OCSVM is the Radial Basis Function (RBF), which is defined in (4) [18].

$$K(x_1, x_2) = exp(-\gamma\left\|x_i - x_j\right\|^2) \tag{4}$$

Where x_i and x_j are vectors in the input space, γ is a kernel parameter that controls the smoothness of the feature mapping in the high-dimensional space. Larger values of γ cause the model's decisions to be more sensitive to small changes in the data, while smaller values of γ results in a smoother model that is less sensitive to noise.

4) Decision Function: Using the Lagrange technique and the RBF kernel function, the decision function in OCSVM is formulated in (5) [17].

$$f(x) = sgn\left(\sum_{i=1}^{N} \alpha_i k(x_i, x_j) - \rho\right) \tag{5}$$

Using the RBF kernel, the (5) becomes (6):

$$f(x) = sgn\left(\sum_{i=1}^{N} \alpha_i exp(-\gamma\left\|x_i - x_j\right\|^2) - \rho\right) \tag{6}$$

D. *Long Short Term Memory (LSTM)*

The prediction model used in this study is Long-Short Term Memory (LSTM). LSTM is a Recurrent Neural Network (RNN) type designed to overcome vanishing gradients and capture long-term dependencies in sequential data [14]. LSTM has the main advantage of cell states that are responsible for storing information and managing the transformation of input memory into output memory [19]. Each LSTM unit has three main components called gates: forget gate, input gate, and output gate. This mechanism determines how information is selectively maintained, updated, or discarded at each step in the time series [14]. In addition, there is also an update gate that helps update the cell state. These four components work together to receive short-term memory, long-term memory, and input at each timestamp, then produce new short-term memory, long-term memory, and output [19]. Mathematically, the mechanism of each gate in LSTM can be explained by the following equations:

1) Input gate: Determines the information received into memory. the input gate is formulated in (7) [19].

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \tag{7}$$

2) Forget gate: Decides which information from previous memory needs to be forgotten. It is formulated in (8) [19].

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f) \tag{8}$$

3) Update gate: Updates the cell state, expressed in two stages [19].

Update cell state, which formulated in (9).

$$c_t = tanh(W_c * [h_{t-1}, x_t] + b_c) \tag{9}$$

Update cell state, formulated in (10).

$$c_t = f_t * c_{t-1} + i_t * c_t \tag{10}$$

4) Output gate: Produces a new output based on the updated cell state [19], which given in (11).

$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o) \tag{11}$$

Hidden state, is expressed in (12).

$$h_t = o_t * \tanh(c_t) \tag{12}$$

Where σ is a function sigmoid, tanh is the hyperbolic activation function, W is the weight for each gate, h_{t-1} is the previous hidden state, x_t is the input at time t, and b is the bias.

The implementation process involves several stages, including the formation of time series data, the design of the LSTM model architecture, and the training of the predictive model.

1) Time Series Data Formation: The data is converted into a sequential format using a sliding window for 12 weeks to capture the temporal patterns present in the sales data, where each window covers 12 weeks of data as input, and the 13th week serves as the target. The raw and clean datasets are processed separately to ensure accurate evaluation.

2) LSTM Model Architecture: The LSTM model in this study consists of three sequential layers followed by a dense output layer. Layer 1, consists of 128 LSTM units, where each unit uses a forget gate to decide which part of the previous cell state *(C_{t-1})* should be retained, an input gate to introduce new information, and an output gate to calculate the hidden state *(h_t)* is handed over to the next layer. A dropout rate of 30% is applied to reduce overfitting, and *the return_sequences = True* parameter ensures that the sequence information is preserved for the next layer.

Layer 2, with 64 LSTM units, continues processing the sequential data using the same LSTM mechanism. The sequence structure is maintained as *return_sequences = True*, and the number of units is reduced to improve the temporal representation.

Layer 3, consisting of 32 LSTM units, produces the final representation of the time series data. Unlike the previous layers, *return_sequences = False* is applied, retaining only the hidden state *(h_T)* to be passed to the Dense layer. Finally, the Dense layer with one unit utilizes the secret state *(h_T)* to generate weekly sales predictions. The model employs the ReLU activation function to introduce non-linearity, uses Mean Squared Error (MSE) as the loss function, and adopts the Adam optimizer to accelerate convergence during backpropagation.

3) Model Training: The model is trained separately on raw and clean data. The training uses the ReLU activation function, while the loss function is Mean Squared Error (MSE). Adam optimizer is used to accelerate model convergence. The training data for the model is transformed into a three-dimensional form with the format (number of windows, window length, number of features) before being fed into the LSTM model. This process ensures the model can optimally learn data patterns from both datasets.

E. Testing and Analysis

The testing and analysis phase involves a weekly sales dataset consisting of 6,435 records, which is used as the population to determine the research sample. To preserve the temporal order of observations and avoid data leakage, the dataset is split into 80% for training and 20% for testing, without any random shuffling. The distribution of data allocation is summarized in Table II.

TABLE II
DISTRIBUTION TO DATA ALLOCATION

Training Data	**Testing data**
4620	1156

The outlier detection process uses the One-Class Support Vector Machine (OCSVM) method with the Radial Basis Function (RBF) kernel. This method learns the normal data distribution based on standardized numerical features, such as weekly sales, temperature, fuel prices, unemployment rates, and CPI. OCSVM uses the nu parameter to determine the proportion of data considered outliers. In contrast, the gamma parameter is automatically set (scale) to adjust the kernel's sensitivity to the data distribution. Data identified as outliers is labelled -1, while inlier data is labelled 1. After detection, data labelled -1 will be removed from the dataset to produce a clean dataset ready for predictive model training.

The data sequence formation process involves converting the time series dataset into a 12-week windowed sequence to capture temporal patterns.

Model testing is done by training a Long Short-Term Memory (LSTM) model on both datasets (raw and clean). Model accuracy is then evaluated using the following assessment metrics:

1) Mean squared Error (MSE) [20]: Used to calculate the average of the squared differences between the actual and predicted values.

$$MSE = \frac{1}{n}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (13)$$

2) Root Mean Squared Error (RMSE) [20]: Used to measure the root mean square error of the difference between the actual and predicted values. The RMSE value is determined from the actual data value (y_i), the expected system value ($\hat{y}_i$), and the number of data points (n).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \quad (14)$$

3) Mean Absolute Error (MAE) [20]: MAE measures the average absolute error between the predicted and actual values without considering the direction (positive or negative). The MAE value provides an overview of the average error size that occurs in each prediction.

$$MAE = \frac{1}{n}\sum_{i=1}^{N}|y_i - \hat{y}_i| \quad (15)$$

III. RESULT AND DISCUSSION

This study analyzes the prediction performance of a Long Short-Term Memory (LSTM) model for weekly sales forecasting after data preprocessing with z-score standardization and outlier detection using the One-Class Support Vector Machine (OCSVM) method.

A. Initial Data Characteristics

The initial dataset consists of 6,435 weekly sales records with multiple numerical features: *Weekly_Sales*, *Temperature*, *Fuel_Price*, *CPI*, and *Unemployment*. Descriptive statistics in Table III show substantial variation in both scale and dispersion, which justifies the use of standardization to balance the influence of each feature during outlier detection and model training.

The standardized dataset ensures that each feature contributes proportionally to the outlier detection process and to the training stability of the LSTM.

B. Outlier Detection and Filtering

The standardized data was then processed using the OCSVM method to identify and filter anomalous records across all numerical features. A total of 701 data points (approximately 10.89%) were identified as outliers and removed to produce a cleaner dataset for subsequent prediction tasks Table IV.

Fig. 2 shows the outlier detection results for each numerical feature in the dataset, including Weekly Sales, Temperature, Fuel Price, CPI, and Unemployment. In each subplot, the red points and shaded areas indicate data instances identified as outliers by the OCSVM model, while the blue points represent normal observations (inliers). This visualization confirms that the OCSVM method successfully captures unusual patterns across all features, helping to remove potential anomalies that could distort the LSTM model's learning process if left untreated. Overall, the plots illustrate that outliers are scattered irregularly across the time span and feature space, underlining the benefit of multivariate outlier detection in this forecasting context.

C. Model Performance Evaluation

The final cleaned and standardized dataset was used to train the LSTM model for weekly sales forecasting. The performance metrics in Table V show that the model achieves a Mean Squared Error (MSE) of 0.03, a Root Mean Squared Error (RMSE) of 0.18, and a Mean Absolute Error (MAE) of 0.11, indicating good alignment between predicted and actual sales.

As visualized in Fig. 3, the predicted values imply that the prediction errors remain low and stable across different measurement scales, indicating that the preprocessing pipeline effectively supports the model in capturing the underlying sales patterns with minimal deviation from actual observations.

TABLE III
DESCRIPTIVE STATISTICS BEFORE AND AFTER PREPROCESSING

Statistic	**Weekly Sales**	**Holiday Flag**	**Temeprature**	**Fuel Price**	**CPI**	**Unemployment**
Raw Count	6435.00	6435.00	6435.00	6435.00	6435.00	6435.00
Raw Mean	1,046,964.88	0.07	60.66	3.36	171.58	8.00
Raw Std	564,366.62	0.26	18.44	0.46	39.36	1.88
Clean Count	5734.00	5734.00	5734.00	5734.00	5734.00	5734.00
Clean Mean	-0.03	0.05	0.00	0.01	0.04	-0.04
Clean Std	0.92	0.22	0.94	0.97	0.99	0.88

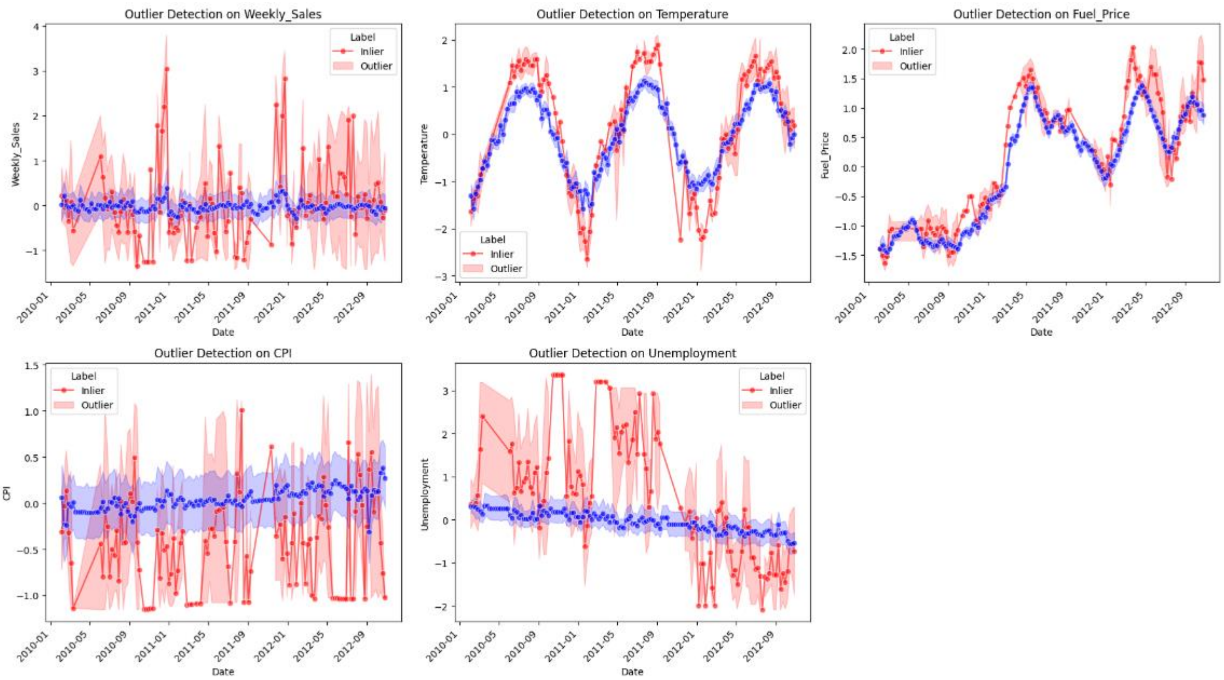

Fig. 2 Outlier detection visualization across multiple features using OCSVM

TABLE IV
SUMMARY OF DETECTION RESULTS USING OCSVM

Description	**Data Count**	**Percentage**
Outlier	701 data	10.89%
Inlier	5.734 data	89.11%

TABLE V
RESULTS OF THE EVALUATION OF RAW AND CLEAN DATA

Metric	**Clean data**
MSE	0.03
RMSE	0.18
MAE	0.11

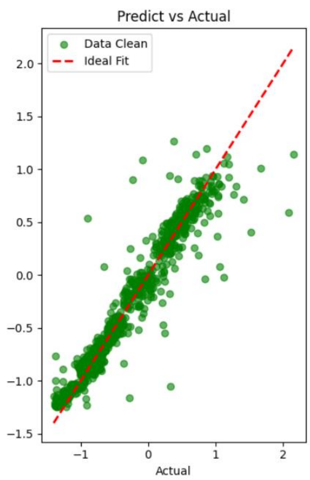

Fig. 3 Predicted vs. actual weekly sales graphic

Compared to the empirical findings in [14], where standardization and imputation techniques were primarily evaluated on univariate datasets such as AirQuality, this study extends the scope by implementing outlier detection using OCSVM on a multivariate, real-world retail sales dataset. While [14] provided a broad overview of preprocessing methods, their evaluation did not explore how outlier detection with OCSVM can be integrated with LSTM to handle real-world retail forecasting tasks involving multiple interrelated features.

IV. CONCLUSION

The results show that LSTM performance for weekly sales prediction significantly improves with data preprocessing, particularly through z-score standardization and OCSVM outlier detection. By filtering anomalous points, noise in the data is reduced, resulting in cleaner inputs and more accurate forecasts, as shown by notable decreases in MSE, RMSE, and MAE. Although Tawakuli et al. (2024) broadly reviewed preprocessing methods, their work did not assess its integration with LSTM nor its effect on multivariate retail forecasting. This study fills that gap by empirically evaluating OCSVM-based preprocessing on real-world sales data, underlining the importance of robust outlier filtering in improving time series prediction accuracy. The results offer valuable implications for practitioners, showing that careful preprocessing can help retailers develop more dependable sales forecasts to inform inventory and business planning. Future research could test other detection methods like Isolation Forests or Autoencoders, apply this pipeline to sectors such as finance or health, and consider including relevant external factors to further enhance forecasting performance and model generalizability.

REFERENCES

[1] S. S. W. Fatima and A. Rahimi, “A Review of Time-Series Forecasting Algorithms for Industrial Manufacturing Systems,” Jun. 01, 2024, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/machines12060380.

[2] “The Analysis of Time Series_ An Introduction with R”.

[3] P. R. Fitrayana and D. R. S. Saputro, “Algoritme Clustering Large Application (CLARA) untuk Menangani Data Outlier,” *PRISMA, Prosiding Seminar Nasional Matematika*, vol. 5, pp. 721–725, 2022, [Online]. Available: https://journal.unnes.ac.id/sju/index.php/prisma/

[4] J. L. Serras, S. Vinga, and A. M. Carvalho, “Outlier detection for multivariate time series using dynamic bayesian networks,” *Applied Sciences (Switzerland)*, vol. 11, no. 4, pp. 1–22, Feb. 2021, doi: 10.3390/app11041955.

[5] L. Melissa Yapanto, S. Muzfirah, Mp. M. Neny Rasnyanti Aras, and Ms. Nurhikmah Sibua, “ANALISIS DATA STATISTIK METODE DAN TEKNIK (STATISTICAL DATA ANALYSIS: METHODS AND TECHNIQUES).”

[6] *KST 2020 : 2020-12th International Conference on Knowledge and Smart Technology : Jan 29 - Feb 1 @Amari Pattaya, Thailand*. IEEE, 2020.

[7] N. Seliya, A. Abdollah Zadeh, and T. M. Khoshgoftaar, “A literature review on one-class classification and its potential applications in big data,” *J Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00514-x.

[8] M. R. Ayyagari, “Classification of Imbalanced Datasets using One-Class SVM, k-Nearest Neighbors and CART Algorithm,” 2020. [Online]. Available: www.ijacsa.thesai.org

[9] M. J. Zaki Wagner Meira Jr, “Data Mining and Analysis: Fundamental Concepts and Algorithms.”

[10] A. De Martino and K. Diki, “Gaussian RBF-kernels via Fock spaces: quaternionic and several complex variables settings,” Aug. 2023, [Online]. Available: http://arxiv.org/abs/2308.14110

[11] E. P. Wahyuddin, R. E. Caraka, R. Kurniawan, W. Caesarendra, P. U. Gio, and B. Pardamean, “Improved LSTM hyperparameters alongside sentiment walk-forward validation for time series prediction,” *Journal of Open Innovation: Technology, Market, and Complexity*, vol. 11, no. 1, Mar. 2025, doi: 10.1016/j.joitmc.2024.100458.

[12] X. Song *et al.*, “Time-series well performance prediction based on Long Short-Term Memory (LSTM) neural network model,” *J Pet Sci Eng*, vol. 186, Mar. 2020, doi: 10.1016/j.petrol.2019.106682.

[13] M. Pala and E. Sefer, “NFT price and sales characteristics prediction by transfer learning of visual attributes,” *Journal of Finance and Data Science*, vol. 10, Dec. 2024, doi: 10.1016/j.jfds.2024.100148.

[14] A. Tawakuli, B. Havers, V. Gulisano, D. Kaiser, and T. Engel, “Survey:Time-series data preprocessing: A survey and an empirical analysis,” *Journal of Engineering Research (Kuwait)*, 2024, doi: 10.1016/j.jer.2024.02.018.

[15] A. Razaki and Y. Herry Chrisnanto, “PENANGANAN OUTLIER PADA METODE ALGORITMA K-NEAREST NEIGHBORS (KNN) DENGAN METODE KERNEL DENSITY ESTIMATION PADA KASUS PENYAKIT DIABETES HANDLING OUTLIERS IN THE K-NEAREST NEIGHBORS (KNN)

ALGORITHM USING KERNEL DENSITY ESTIMATION IN DIABETES CASES," *Journal of Information Technology and Computer Science (INTECOMS)*, vol. 7, no. 4, 2024.

[16] B. Hakim, "Analisa Sentimen Data Text Preprocessing Pada Data Mining Dengan Menggunakan Machine Learning," *JBASE - Journal of Business and Audit Information Systems*, vol. 4, no. 2, Aug. 2021, doi: 10.30813/jbase.v4i2.3000.

[17] S. S. Todkar, V. Baltazart, A. Ihamouten, X. Dérobert, and D. Guilbert, "One-class SVM based outlier detection strategy to detect thin interlayer debondings within pavement structures using Ground Penetrating Radar data," *J Appl Geophy*, vol. 192, Sep. 2021, doi: 10.1016/j.jappgeo.2021.104392.

[18] M. J. Zaki Wagner Meira Jr, "Data Mining and Analysis: Fundamental Concepts and Algorithms."

[19] K. E. ArunKumar, D. V. Kalaga, C. Mohan Sai Kumar, M. Kawaji, and T. M. Brenza, "Comparative analysis of Gated Recurrent Units (GRU), long Short-Term memory (LSTM) cells, autoregressive Integrated moving average (ARIMA), seasonal autoregressive Integrated moving average (SARIMA) for forecasting COVID-19 trends," *Alexandria Engineering Journal*, vol. 61, no. 10, pp. 7585–7603, Oct. 2022, doi: 10.1016/j.aej.2022.01.011.