ISSN 2723-6471

Application of Random Data Splitting Technique for Time Series Modeling Using Feedforward Neural Network and Support Vector Regression Models

B. R. P. M. Basnayake[1,*], N. V. Chandrasekara[2]

[1,2]*Department of Statistics and Computer Science, Faculty of Science, University of Kelaniya, 11600, Sri Lanka*

[1]*Department of Statistics and Computer Science, University of Peradeniya, 20400, Sri Lanka*

(Received: October 25, 2025; Revised: December 15, 2025; Accepted: March 22, 2026; Available online: May 3, 2026)

Abstract

In predictive modeling, data partitioning mainly involves dividing the dataset into a training set for learning model parameters and a testing set for assessing generalizability through predictive performance. In time series forecasting, non-random data splitting is commonly used where initial observations are used for training, a subsequent portion for validation and latest observations are utilized for testing. However, the effectiveness of random data partitioning where observations are randomly distributed into training, validation and testing subsets remains underexplored within the context of time series data. This study investigates the suitability of random data partitioning in time series forecasting using both stationary and non-stationary simulated datasets, as well as real-world data. Feedforward neural network (FFNN) and Support vector regression (SVR) models were implemented, with model performance optimized through systematic trial-and-error hyperparameter tuning. Under random data-splitting approach, 30 different training and testing subsets are generated to assess the stability and robustness of model performance across different sample compositions. Random data partitioning is implemented at the level of constructed supervised learning instances, where each instance consisted of lagged input variables and their corresponding target values, forming distinct input–output pairs. The allocation of these instances to training and testing subsets is carried out entirely at random, without preserving sequential ordering or grouping temporally adjacent observations, while strictly maintaining original input–output correspondence within each constructed instance. The findings indicate that, for both simulated and real-world datasets, random data splitting resulted in improved predictive performance of both models, yielding lower error metrics compared to non-random splitting. These results suggested that random data splitting can enhance generalization and forecasting performance in time series applications with appropriate models. The study provides valuable empirical evidence on data partitioning strategies, supporting researchers and practitioners in making more informed decisions regarding model evaluation and selection in time series forecasting tasks.

Keywords: Feedforward Neural Network, Non-Random Data Split, Random Data Split, Support Vector Regression, Time Series Data

1. Introduction

Time series modeling and forecasting have long been a fundamental focus of academic research, playing a crucial role in various applications, including finance, economics, biological sciences, medicine and etc. For several decades, time series forecasting has predominantly utilized classical methods, such as Autoregressive integrated moving average (ARIMA), Seasonal ARIMA (SARIMA), time series regression and exponential smoothing techniques, due to their advantageous properties in model fitting and interpretability [1], [2], [3]. However, such models are limited in their ability to capture the nonlinear patterns commonly observed in real-world data [4], [5], [6], [7]. Hence, several nonlinear parametric time series models have been proposed in the literature, such as Bilinear (BL), Sign Autoregressive (SAR), Nonlinear Autoregressive (NAR), Nonlinear Moving Average (NMA), Threshold Autoregressive (TAR) and Smoothing Transition Autoregressive (STAR) models [4], [6], [8]. These models are applied within specific datasets and problem domains and the predefined structure of these models limits the effectiveness of parametric nonlinear approaches, given the presence of various nonlinear patterns. Moreover, such rigid structures restrict the development of dynamic models that can flexibly adapt to evolving patterns in time series data.

***Corresponding author: B. R. P. M. Basnayake (pavithrab@sci.pdn.ac.lk)**
DOI: https://doi.org/10.47738/jads.v7i2.1216

Consequently, selecting an appropriate nonlinear model for a particular dataset becomes a complex task due to the numerous potential models and parameters available. Unlike the aforementioned models, machine learning models are nonparametric, data-driven approaches that can identify nonlinear data patterns without relying on prior assumptions regarding the underlying relationships within a given problem [4], [9]. Due to the wide-ranging nature of time series problems across different fields, various machine learning models have been developed for forecasting purposes. The Feedforward Neural Network (FFNN) and Support Vector Regression (SVR) models are widely recognized as primary models for time series forecasting [4], [5], [6], [7], [10].

In predictive modeling, data partitioning involves dividing the available dataset into two distinct subsets: a training set, used to learn the model parameters and a testing set, reserved for evaluating the model's final predictive performance and generalizability [4], [5], [6], [7], [8]. In many modeling approaches, particularly those involving complex architectures such as neural networks, a third subset, known as the validation set, is also introduced. This validation set plays a crucial role in mitigating issues of overfitting or underfitting. Data partitioning is fundamental for assessing a model's ability to generalize to unseen data. Without proper partitioning, there is a risk of overfitting, where a model performs well on the training data but fails to make accurate predictions on new, unseen observations. This indicates the reliability of the evaluation, as using the training set for performance assessment offers no assurance of the model's ability to generalize to future data. Further, it is important to ensure unbiased model selection and evaluation. By holding out a portion of the data for testing, researchers can measure how well the trained model performs on data it has not previously seen. Furthermore, a validation set helps in detecting potential underfitting or overfitting, thereby improving overall model performance. There are several commonly used strategies for data partitioning in predictive modeling. One approach is random splitting, where data points are randomly assigned to training, validation and test subsets, ensuring that each partition represents the overall data distribution. Another widely used method in time series analysis is time-based splitting, also referred to as non-random splitting, where the initial observations are used for training, a subsequent portion is designated for validation and the latest observations are reserved for testing, preserving the temporal order of the data.

Most studies on machine learning based models for time series forecasting have relied on the non-random data splitting procedure, where the initial observations are allocated for training, followed by validation (in some cases) and the final set of observations is reserved for testing [4], [5], [6], [7]. The literature has given limited attention to the effectiveness of employing random data splitting methods for training, testing and validating time series data using appropriate machine learning models. Non-random data splitting can lead to models adapting to biased patterns during training and this adaptation may reduce the models' ability to generalize their performance to unseen data, increasing the likelihood of overfitting. For classical statistical models such as ARIMA and SARIMA, random data splitting is not feasible because these methods explicitly rely on the sequential structure of the observations. Their parameter estimation and forecasting performance depend on preserving the temporal order, as past values are used directly to model future outcomes. Randomly shuffling or partitioning the data would disrupt these dependencies, invalidate the underlying model assumptions and ultimately lead to unreliable parameter estimates and forecasts.

In contrast, for the machine learning models such as FFNN and SVR, temporal information can be incorporated indirectly through constructed input–output pairs, where lagged observations are treated as explanatory variables. Once these input–output relationships are defined, each data instance, represented as a row containing the input variables and the corresponding output, can be treated independently, provided that the input–output correspondence is preserved. As a result, random data partitioning does not inherently violate the modeling framework of these methods, allowing flexibility in data splitting strategies without undermining the validity of parameter estimation or predictive performance. This flexibility allows the use of random splitting strategies, where training and testing sets are drawn from different portions of the series while still retaining the integrity of the predictor–response mapping. As a result, FFNN and SVR can be trained on randomized subsets of the data, which helps reduce dependence on a specific temporal sequence and enables a more robust evaluation of the model's ability to generalize across unseen observations. To evaluate the applicability of the random data split procedure for time series data, this study applies FFNN and SVR, which are static models that do not inherently capture temporal dependencies.

Unlike recurrent models, SVR and FFNN treat each input as an independent feature vector, making them suitable for assessing whether breaking the sequential order of data through a random split affects forecasting performance. By

comparing the results from the random split, where the input and corresponding output pairs remain aligned and the non-random split, this study examines whether maintaining the temporal sequence in data splitting is essential for these models or if a random split can serve as an alternative approach with higher predictive performance. Hence, the primary objective of this study is to systematically examine the applicability of the random data split procedure for time series data using FFNN and SVR models and to compare their performance with the non-random data split methodology. This is achieved through an extensive experimental investigation involving both simulated datasets and real-world time series data to comprehensively assess their forecasting effectiveness under random and non-random data partitioning strategies. This study builds upon our prior research that demonstrated the effectiveness of the random data split procedure over the non-random split for stationary time series data when using the FFNN model [8].

Our prior study work was limited to simulated stationary datasets and focused exclusively on FFNN. In contrast, the present study extends the analysis to both stationary and non-stationary simulated datasets, while also incorporating real-world data and evaluating the performance of both FFNN and SVR models. Hence, the novelty of this study lies in its comprehensive evaluation of random versus non-random data splitting strategies using both simulated and non-stationary real-world time series datasets across a wide range of time resolutions, including hourly, daily, weekly, monthly, quarterly and annual data. Furthermore, this study focuses on FFNN and SVR models to assess their ability to capture accurate dynamics under both random and non-random data partitioning strategies for time-series data. In addition, the study emphasizes the role of hyperparameter tuning in optimizing model performance, providing empirical evidence on the effectiveness and suitability of random data splitting approaches in time series forecasting. The main contributions of this research are: (1) it provides the comprehensive empirical evaluation of the effects of random data splitting strategy on forecasting performance in machine learning models using FFNN and SVR, with both simulated and real-world time series data; and (2) it provides valuable guidance on effective data partitioning strategies for time series forecasting, emphasizing the importance of hyperparameter optimization in enhancing the predictive performance of FFNN and SVR models. The findings contribute to a deeper understanding of data partitioning strategies in time series forecasting, offering practical implications for model selection and validation in real-world applications.

The structure of the paper is outlined as follows: The next sections present a detailed explanation of the literature review and methodology employed. Subsequently, an analysis of the results and discussion is presented. The paper concludes by summarizing the key findings and final remarks.

2. Literature Review

Numerous studies have applied FFNN and SVR models for predictive modeling, mainly employing non-random data splits for time-series data to preserve the chronological structure required for accurate forecasting, while random splits have been used for non-time-series data where observations are assumed independent of time [4], [7], [11]. For instance, an empirical evaluation was conducted to assess the performance of FFNN in relation to the number of input nodes, hidden nodes and sample size for nonlinear time series forecasting [4]. The study involved 30 replications on eight simulated series of BL, SAR, NAR, NMA, TAR and STAR based on nonlinear time series models. Each series consisted of 480 data points, with the final observations divided into three test sets (20, 40 and 80 observations) to represent different forecasting horizons. Non-random splitting was employed to preserve the temporal order of the series and the use of three separate test sets allowed the study to evaluate model performance across short-, medium- and long-term forecasting horizons, providing a more comprehensive assessment of the FFNN's predictive capabilities under varying forecast lengths. Performance measures of Mean Squared Error (MSE) and Mean Absolute Percentage Errors (MAPE) were computed for each horizon and the results demonstrated that FFNNs outperformed ARIMA models in the prediction of nonlinear time series.

Additionally, it was found that the number of input nodes had a significantly greater impact on the neural network model's performance than the number of hidden nodes. Further, a larger sample size helped mitigate overfitting and FFNN models with one or two hidden nodes in a single hidden layer exhibited the lowest error values in their study. Similarly, another study focused on finding the impact of various data preprocessing techniques, including deseasonalization and detrending, on FFNN forecasting performance with simulated and real-world time series data [7]. For simulated data, a total of 228 observations were generated according to Equation 10 with different noise levels.

The initial 216 observations were utilized for model selection and parameter estimation, while the final 12 observations (non-random split) served as the test set for forecasting evaluation and comparison. Similarly, for the selected real-world datasets, the last 12 months of data were reserved as the test set for forecasting assessment, with the initial observations used for model selection and estimation under a non-random data split. In both cases, the non-random split was adopted to maintain the chronological order of the series, ensuring that past information was used to predict future values. The study's findings indicated that neural networks face challenges in accurately capturing seasonal and trend components when trained on unprocessed raw data. However, applying either detrending or deseasonalization significantly reduced forecasting errors and the combination of both techniques proved to be the most effective preprocessing approach with the lowest forecasting error values, as measured by RMSE, MAE and MAPE. An empirical study evaluated the effectiveness of various performance metrics, including 5-fold cross-validation, leave-one-out, radius bound, Vapnik measure, span bound, regularized risk and training error, in optimizing SVR hyperparameters efficiently and accurately using the Auto imports database and the Boston housing data that are not time-dependent [11].

Among these methods, only k-fold cross-validation, leave-one-out and span bound identified optimal models with the lowest error values due to their capacity for better generalization, avoidance of overfitting and robustness across different datasets. The analysis of this study was conducted using more than ten different training and test splits and the better-performing SVR hyperparameter values were determined based on the MSE observed in the test data. Another study evaluated the performance of Support Vector Machines (SVM) with respect to the gamma parameter across different kernel functions [10]. The analysis focused on how variations in the gamma value influenced the classification efficiency of SVMs when applied to diverse non-time series datasets obtained from the UC Irvine machine learning repository. For both the training and testing phases, each dataset was randomly partitioned into training and testing subsets, as there were no temporal dependencies that needed to be maintained. The findings indicated that the effect of gamma on classification accuracy varied across kernels. In particular, polynomial and Sigmoid kernels were more sensitive to changes in gamma depending on the dataset used, whereas the radial basis function kernel demonstrated comparatively stable performance, with only minor fluctuations in accuracy across different gamma values.

In a separate study, a simulation was conducted to assess and compare the performance of various machine learning models, including k-nearest neighbors, FFNN, Naïve Bayes, decision trees, random forests, gradient-boosting trees and logistic regressions for linear and non-linear time series prediction [6]. The study employed eight linear and nonlinear models to simulate the data and the complexity of those models was further increased by incorporating a compound Poisson jump process and additional noise into the simulated data. A non-random data split procedure was applied, with the initial 80% of the data used for training the models and the remaining 20% for testing. The findings of the study indicated that FFNN models outperformed the other models in terms of predictive accuracy. A comparative analysis was conducted to evaluate the performance of Long Short-Term Memory (LSTM) networks, gated recurrent units (GRU), simple RNN (SRNN) and FFNN using Adaptive Moment Estimation (ADAM) and Root Mean Square Propagation (RMSprop) optimization algorithms for forecasting exchange rates of EURO/USD, GBP/USD, USD/JPY and USD/CHF [12].

The models were trained on a selected training set and predictions were generated on the test set using a non-random split procedure in their study and the results indicated that, in terms of training loss, LSTM and GRU exhibited slightly better performance than FFNN and SRNN across all considered time series data. However, in terms of forecasting performance, FFNN outperformed other network architectures. Our prior work presented a comprehensive simulation to evaluate the performance of FFNN models using the random data split procedure for stationary time series data [8]. In the study, eight nonlinear models (BL, SAR, NAR, NMA, TAR and STAR) were employed to generate multiple time series, each representing different characteristics of the data. The complexity of the models was increased by integrating Poisson processes with varying jump sizes. For each selected model, 30 replications were produced using different initial random values assigned to the error term. The dataset was partitioned using a random split strategy, assigning 80% for training, 10% for validation and the remaining 10% for testing purposes. FFNN models were fitted using inputs from past observations and moving average values and each model was trained 30 times to provide a statistically robust evaluation of performance by averaging the outputs. The results of the study demonstrated that the

 ISSN 2723-6471

FFNN models performed effectively with the random data split procedure, yielding lower minimum error values of MSE, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and MAPE. This was further substantiated through graphical comparisons between the actual test values and the average forecasted observations. In another of our prior investigation, the work explored the effectiveness of using neural networks with various optimization algorithms for predicting major currencies of CAD, AUD, EURO, CHF, GBP, JPY, USD and SGD against LKR using FFNN, LSTM and Generalized Regression Neural Network (GRNN) models [5].

To evaluate the predictive capability of each model, error metrics of MSE, MAE, MAPE and RMSE were used. The dataset was partitioned non-randomly into three subsets: the training period spanned from January 1, 2008, to November 16, 2021; the validation period extended from November 17, 2021, to January 7, 2022; and the testing period covered January 8, 2022, to February 28, 2022. The input data included historical lagged values and moving average indicators for various periods. The study reported superior predictive performance of FFNNs, TDNNs and LSTM models compared to GRNNs in the context of exchange rate forecasting. Notably, the FFNN with the RMSprop algorithm produced the lowest error for AUD/LKR, CHF/LKR, EURO/LKR and JPY/LKR. For GBP/LKR, the FFNN with ADAM was most effective, while TDNN with RMSprop was superior for CAD/LKR. The LSTM model optimized with ADAM demonstrated the highest accuracy for forecasting SGD/LKR, while the optimization algorithm using RMSprop yielded the best performance for USD/LKR.

The literature review highlights previous studies that implemented the random data split procedures for non-time series data and the non-random data split procedures for time series data. Existing studies based on time series forecasting have predominantly relied on non-random data partitioning methods, where model training and testing are performed on sequentially divided datasets. Conversely, random data splitting has been commonly applied only in non-time series contexts to evaluate model generalization. To date, no comprehensive investigation has systematically assessed the applicability and impact of random data splitting procedures on time series forecasting using FFNN and SVR models, particularly by comparing them with the classical non-random splitting strategy. This study addresses this gap by providing empirical evidence on the effectiveness of random data splits in capturing diverse data patterns and improving predictive accuracy across both simulated and real-world time series datasets. Hence, the research offers new insights into data partitioning strategies for time series modeling, contributing to more robust model evaluation and selection practices.

3. Methodology

An experiment was conducted to evaluate the feasibility of applying the random data splitting methodology to stationary and non-stationary time series data. For this purpose, both simulated and real-world datasets were utilized, where the performance of FFNN and SVR models under random data partitioning was assessed and compared against the classical non-random data splitting approach. Figure 1 illustrates the overall methodological framework of the study.

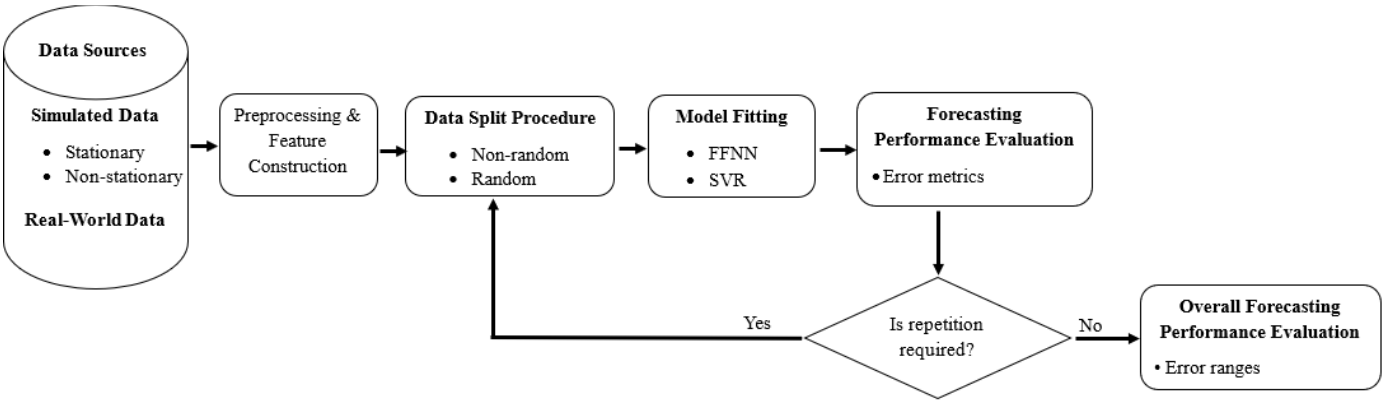

Figure 1. Flowchart of the methodology

3.1. Data

This study includes data spanning various time resolutions, including simulated and real-world data with hourly, daily, weekly, monthly, quarterly and annual datasets. Incorporating data with various time resolutions is essential for a comprehensive evaluation of the proposed methodology. Different time components capture unique patterns and characteristics in time series data, such as short-term fluctuations, seasonal variations and long-term fluctuations. Using a range of temporal resolutions ensures that the model's performance is assessed across diverse scenarios, providing insights into its adaptability and robustness.

Stationary nonlinear univariate time series y_t were simulated using eight different nonlinear models of Bilinear (BL), Sign Autoregressive (SAR), Nonlinear Autoregressive (NAR), Nonlinear Moving Average (NMA), Threshold Autoregressive (TAR) and Smoothing Transition Autoregressive (STAR) models specified in Equation 1 to Equation 8 [4], [6], [8]. These models are well-established in the literature for capturing diverse structural properties of real-world time series and effectively represent complex nonlinear dependencies [4], [6], [13], [14], [15]. The selected formulations include autoregressive (AR) processes, moving average (MA) mechanisms, as well as hybrid specifications combining AR and MA terms. This range of models was chosen to reflect heterogeneous nonlinear dynamics frequently observed in empirical datasets. For each specification, time series of length (L) of 2500 was generated, with 30 independent replications conducted under distinct random values to incorporate variability in the innovation term (ε_t) . The ε_t was assumed to be independently and identically distributed according to a standard normal distribution, $\varepsilon_t \sim N(0,1)$.

BL model 1
$$y_t = 0.7y_{t-1}\varepsilon_{t-2} + \varepsilon_t \quad (1)$$

BL model 2
$$y_t = 0.4y_{t-1} - 0.3y_{t-2} + 0.5y_{t-1}\varepsilon_{t-2} + \varepsilon_t \quad (2)$$

SAR model
$$y_t = sign(y_{t-1}) + \varepsilon_t$$
$$\text{Where} sign(x) = 1 \;\; if \; x > 0$$
$$sign(x) = 0 \quad if \; x = 0$$
$$sign(x) = -1 \;\; if \; x < 0 \quad (3)$$

NAR model
$$y_t = \frac{0.7|y_{t-1}|}{|y_{t-1}| + 2} + \varepsilon_t \quad (4)$$

NMA model
$$y_t = \varepsilon_t - 0.3\varepsilon_{t-1} + 0.2\varepsilon_{t-2} + 0.4\varepsilon_{t-1}\varepsilon_{t-2} - 0.25\varepsilon_{t-2}^2 \quad (5)$$

TAR model
$$y_t = 0.9y_{t-1} + \varepsilon_t \text{ for } |y_{t-1}| \le 1$$
$$= -0.3y_{t-1} - \varepsilon_t \text{ for } |y_{t-1}| > 1 \quad (6)$$

STAR model 1
$$y_t = 0.8y_{t-1} - 0.8y_{t-1}[1 + \exp(-10y_{t-1})]^{-1} + \varepsilon_t \quad (7)$$

STAR model 2
$$y_t = 0.3y_{t-1} - 0.6y_{t-2} + (0.1 - 0.9y_{t-1} + 0.8y_{t-2})[1 + \exp(-10y_{t-1})]^{-1} + \varepsilon_t \quad (8)$$

In the subsequent stage, each generated time series was polluted by incorporating a compound Poisson jump process (k_t), thereby producing a new series (z_t), as expressed in Equation 9 [6].

$$z_t = y_t + k_t \quad (9)$$

where $k_t = \sum_{i=1}^{N_t} D_i$, $D_i \sim N(0, s_p^2)$, $N_t \sim P(r_p)$. The parameter s_p is the jump magnitude, with values set at 0.1, 1 and 10, while r_p denotes the jump rate given by $r_p = \frac{L}{10}$ [6]. A small magnitude, such as 0.1, corresponds to minor fluctuations and reflects settings characterized by relatively stable dynamics. The intermediate setting of 1 reflects moderate jumps that balance between stability and volatility, thereby representing more typical real-world shifts. The large value of 10 corresponds to extreme jumps, which are useful for testing the robustness of the model under highly volatile or shock-driven conditions. By covering this range from low to high, the analysis ensures that the behavior of the model can be evaluated under diverse scenarios, rather than being restricted to a narrow band of jump magnitudes. Subsequently, the following non-stationary multiplicative model was used to generate the simulated series [7].

$$y_t = T_t S_t + E_t \quad (10)$$

where the linear trend is given by $T_t = 100 + 0.6t$, the seasonal index is denoted by S_t and the error term is $E_t \sim N(0, \sigma^2)$. The impact of noise levels was analyzed by considering three error term variance levels of 0.1, 1 and 10 and for each series a total of 228 data points were generated based on Equation 10. Each simulated series was generated 30 times using different initial random values for the error term and this approach ensures a comprehensive evaluation of model performance by capturing variability in the data and assessing the robustness of the models across different error terms.

The real-world datasets considered in this study include a diverse range of time series data, all of which are freely accessible through the repositories mentioned below. These include annual average surface air temperature in Celsius for the United States from 1901 to 2022 with 122 observations obtained from the World Bank Climate Change Knowledge Portal, annual life expectancy at birth, sourced from the World Bank data catalog from 1960 to 2021, resulting in 62 observations, quarterly data from the Federal Reserve Bank of St. Louis for global wheat price in USD per Kilogram from January 1990 to January 2024 with 137 observations, monthly cinnamon import data for the USA in Metric Tons, spanning from May 2005 to January 2024, with a total of 229 observations, monthly crude oil (petroleum) price in dollars per barrel from the World Bank data catalog, covering the period from February 2009 to February 2024, with 181 observations, gold price per troy ounce were sourced from World Gold Council, with weekly data recorded from January 2010 to June 7, 2024 with 754 observations, daily exchange rates for EURO/LKR, USD/LKR and JPY/LKR were obtained from the Yahoo Finance database, spanning January 1, 2008, to February 28, 2022, with 5,471 observations and hourly electricity price data for Spain, measured in euros per megawatt-hour (€/MWh), were sourced by the Spanish TSO Red Eléctrica de España for the period 2015-2019 with 35,064 observations. The sample sizes of the real-world datasets range from a minimum of 62 observations to a maximum of 35,064 observations.

3.2. Research Design

In this study, FFNN and SVR models were utilized to evaluate the performance of random and non-random data split procedures for time series data. Consistent with prior studies, five input features were selected for model fitting with the stationary data: lag 1 (the immediately preceding value), lag 2 (the value preceding lag 1) and moving averages with window sizes of 2 (MA2), 4 (MA4) and 8 (MA8) [4], [6]. Similarly, for modeling the non-stationary simulated data, ten lagged inputs were considered based on prior literature: 1–4, 12–14, 24, 25 and 36. Lags of 12, 24 and 36 were specifically included to capture the seasonal patterns inherent in monthly data [7]. This fixed selection of inputs ensures that any observed differences in model performance can be attributed to the data partitioning strategy rather than variations in input design. For the real-world data, the fitted FFNN and SVR models incorporated lagged data and appropriate moving average indicators as input variables, selected based on their statistical significance determined through the Spearman correlation coefficient and autocorrelation function [12], [16], [17]. Several combinations of input variables were explored to identify the most effective predictors. The inclusion of lagged data is essential in time series modeling as it captures temporal dependencies and underlying patterns, enabling the model to learn from past observations and improve predictive accuracy. Similarly, moving averages served as effective smoothing techniques that filter out short-term fluctuations and highlight long-term trends, thereby enhancing the model's ability to detect meaningful patterns within the data.

3.2.1. Feedforward Neural Network (FFNN)

In the FFNN, the term "feed-forward" refers to the propagation of neuron outputs from one layer to the next in a unidirectional (forward) manner. The network architecture comprises an input layer, one or more hidden layers and an output layer and each layer consists of interconnected input neurons, hidden neurons and an output node. During the processing of input data (x_t) at a specific time (t), hidden neurons are activated through the activation function (g_h), where h_t represents the output (or activation) of the hidden layer at time t, W_h^T represents the weight matrix and b_h is the bias, as described in Equation 11 [4], [7], [12], [17], [18].

$$h_t = g_h\left(W_h^T x_t + b_h\right) \tag{11}$$

The predicted value $(\hat{y}_t)$ is computed by applying an appropriate output activation function (g_y) to the sum of the weighted activations (W_y^T) and the bias in the output layer, as illustrated in Equation 12.

$$\hat{y}_t = g_y\left(W_y^T h_t + b_y\right) \tag{12}$$

In time series modeling, data is typically divided sequentially, with the initial observations used for training and the final observations reserved for testing. In some cases, a validation set is also included, depending on the size of the

dataset and the specific modeling strategy. Incorporating a validation set is particularly beneficial when the dataset is sufficiently large, as it enables the detection of underfitting or overfitting during the model development process. Common practice in time series modeling involves allocating a larger portion of the data, typically 70%, 80%, or 90% for training. Careful attention must be paid to how data is partitioned, with the aim of maximizing the number of observations used for training to provide the model with the most comprehensive learning experience. The remaining data is then divided between validation and testing, ensuring that model performance can be reliably assessed on unseen observations [4], [5], [6], [7]. In contrast, for non-time series data, the division into training, validation and test sets is often performed randomly. In this study, the non-random splitting technique assigns the initial 80% of the observations for model training, followed by the subsequent 10% for validation and reserves the final 10% for testing, maintaining the sequential order of the data [19], [20], [21]. In the random split approach, 80% of the data is randomly chosen for training, while the remaining 20% is evenly distributed between validation and testing sets. To ensure robustness and account for variability in the results, each FFNN model, corresponding to the set of tuned hyperparameters, was retrained 30 times under both random and non-random data splitting strategies. Forecasting was then performed for each retrained model and the average of the forecasted values was compared with the corresponding test values [4], [6], [8]. Moreover, to ensure the robustness and reliability of the model evaluation under the random data split approach, 30 different random seed values were employed to partition the dataset into training, validation and testing subsets. This repeated training process allowed for a more comprehensive evaluation of model performance across different data splits.

The performance of the FFNN model is influenced by various hyperparameters that play a critical role in forecasting performance. Hyperparameter tuning was conducted using a trial-and-error approach within predefined parameter ranges. Within these ranges, all possible combinations of hyperparameters were systematically evaluated and the configuration yielding the best forecasting performance was selected [4], [5], [7], [12]. The FFNN model was trained using the ADAM optimization algorithm, with the number of hidden layers (L_h) and hidden neurons (N_{hidden}) fine-tuned within a range of 1 to 4. Activation functions (AF) for both the hidden and output layers, were adjusted using hyperbolic tangent (Tanh), Sigmoid, Softsign and linear transfer functions. The learning rate (η) was optimized by adjusting parameter values within the range of 0 to 0.9, number of epochs (N_{epochs}) was fine-tuned within the range of 100 to 1000, while batch sizes (B_{size}) were varied between 16, 32, 64, 128 and 256 with the dropout rate (P_{drop}) ranging from 0 to 0.9. The FFNN models were trained using MAE as the loss function, as it offers a robust measure of deviation that is less sensitive to outliers compared to MSE and learning curves were examined to evaluate potential underfitting or overfitting during the training process [5], [12].

3.2.2. Support Vector Regression (SVR)

SVR is a supervised learning method that adapts the core concepts of SVM to address regression problems [10], [11], [22]. It aims to find a function that approximates the underlying relationship between input features and the target variable by mapping data into a high-dimensional feature space using a kernel function. The goal of SVR is to minimize the error within a specified margin while maintaining model simplicity. The solution of the SVR model is obtained by minimizing the objective function presented in Equation 13 [10], [11].

$$\frac{1}{2}(W^T W) + C\sum_{i=1}^{l}\xi_i \tag{13}$$

$y_i(W^T\phi(x_i) + b) \geq 1 - \xi_i$, W represents a normal unit vector that is orthogonal to the boundary margin, (x_i, y_i) is the training set pairs from $i = 1, \ldots, l$, ε is a threshold parameter that defines the width of the ε -tube, which controls how much error is tolerated in the model, ξ deviation greater than ε, where it is a slack variable that quantifies how much a training example deviates from the ε tube, C is the cost parameter that controls the trade-off between a smooth decision boundary and classifying training points correctly, b represents a slack variable and training sets (x_i) are transformed into a higher-dimensional space through the kernel function of ϕ.

The SVR analysis utilizes a random data split with multiple seed values, allocating 90% of the data for training and 10% for testing, ensuring that features and target values remain correctly paired after splitting and hyperparameter tuning to determine the optimal parameters by selecting those that minimize error metrics on the test data sets. This approach allows for robust performance evaluation, accounting for both hyperparameter optimization and model stability across different random data partitions. For hyperparameter tuning, the optimal epsilon value (ε) was first selected from a range between 0.001 and 0.9, followed by tuning the cost parameter (C) within the range of 2^{-2} to 2^{15} and fine-tuning the gamma parameter (γ) over the range of 2^{-15} to 2^{3}; additionally, various kernel functions, including linear, radial, polynomial and Sigmoid, were tested to identify the best-performing configuration [10], [11], [23].

 ISSN 2723-6471

3.2.3. Data Split Procedure

Consider a time series $y_1, y_2, \ldots, y_N$, where y_t denotes the target value at time t and the predictors consist of its previous values (lags). The parameter p represents the number of lagged terms used as input features, $X_t = [y_{t-1}, y_{t-2}, \ldots, y_{t-p}]$ and if the dataset consists of pairs (X_t, y_t) for $t = p+1, \ldots, N$ with a total size of $N' = N - p$, then the dataset is $D = \{(X_{p+1}, y_{p+1}), (X_{p+2}, y_{p+2}), \ldots, (X_N, y_N)\}$.

Using the predefined split ratio (r), the training set size is determined as $N_{train} = r \times N'$ and while the test set size is calculated as $N_{test} = N' - N_{train}$.

In non-random data split the train set (D_{train}) and test set (D_{test}) is:

$$D_{train} = \{(X_{p+1}, y_{p+1}), (X_{p+2}, y_{p+2}), \ldots, (X_{p+N_{train}}, y_{p+N_{train}})\} \quad (14)$$

$$D_{test} = \{(X_{p+N_{train}+1}, y_{p+N_{train}+1}), \ldots, (X_N, y_N)\} \quad (15)$$

The figure 2 depicts the transformation of the original time series into supervised learning instances, where each row represents a complete input–output pair formed using p lagged observations to predict the subsequent value.

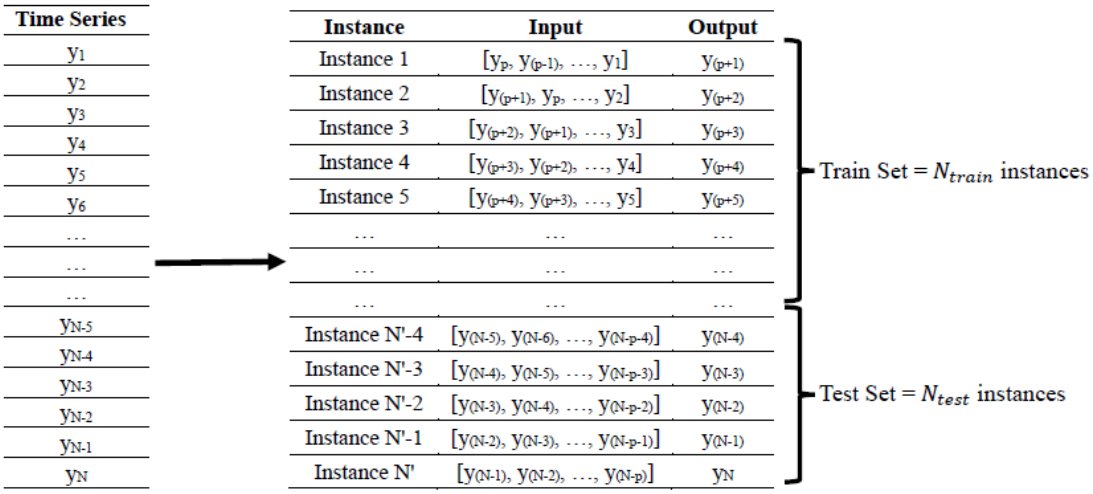

Figure 2. Graphical illustration of the non-random (sequential) data partitioning strategy

The upper portion of the table corresponds to the first N_{train} instances, which are allocated to the training set under the non-random (sequential) partitioning strategy. The lower portion contains the remaining N_{test} instances, which form the test set. This visual representation emphasizes that, in the non-random split, the partitioning follows the original temporal order of the data, where earlier constructed instances are used for model estimation, while later instances are reserved for out-of-sample evaluation.

If π is a function that maps the original indices of the dataset to a randomly shuffled order before splitting it into training and testing sets, where $\pi(i)$ gives the new position of index i after random shuffling. The set $\{i_1, i_2, \ldots, i_{N'}\}$ is a permutation of $\{1, 2, \ldots, N'\}$. Applying π to the dataset results:

$$\pi : \{1, 2, \ldots, N'\} \rightarrow \{i_1, i_2, \ldots, i_{N'}\} \quad (16)$$

Then the randomly shuffled dataset (D') is:

$$D' = \{(X_{\pi(1)}, y_{\pi(1)}), (X_{\pi(2)}, y_{\pi(2)}), \ldots, (X_{\pi(N')}, y_{\pi(N')})\} \quad (17)$$

In random data split the train and test set is:

$$D'_{train} = \{(X_{\pi(1)}, y_{\pi(1)}), (X_{\pi(2)}, y_{\pi(2)}), \ldots, (X_{\pi(N_{train})}, y_{\pi(N_{train})})\} \quad (18)$$

$$D'_{test} = \{(X_{\pi(N_{train}+1)}, y_{\pi(N_{train}+1)}), (X_{\pi(N_{train}+2)}, y_{\pi(N_{train}+2)}), \ldots, (X_{\pi(N')}, y_{\pi(N')})\} \quad (19)$$

 ISSN 2723-6471

Each observation retains the relationship between the lagged input vector X_t and its corresponding target y_t, ensuring that the input–output structure of the time series is preserved despite random shuffling. To further clarify the mechanism of random partitioning, an illustrative example is presented in figure 3.

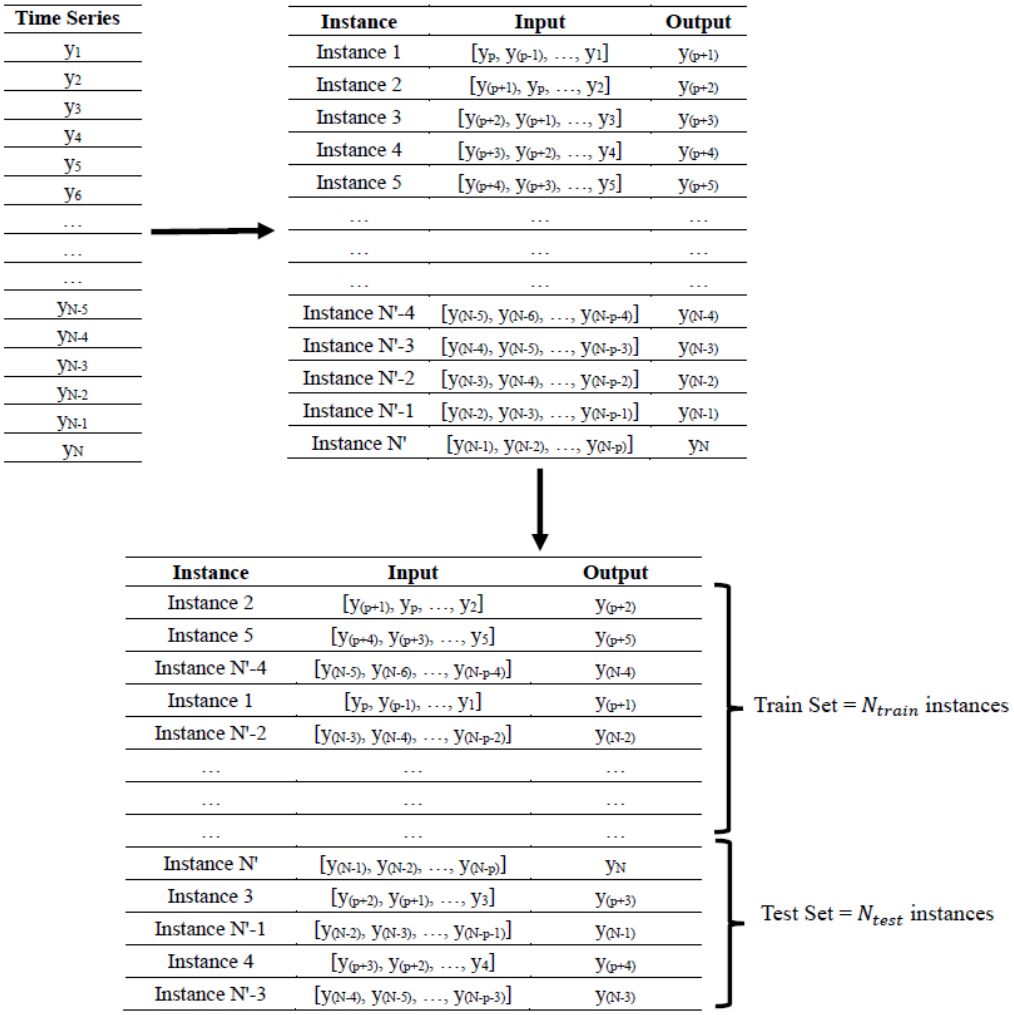

Figure 3. Graphical illustration of the random data partitioning strategy

It displays the dataset after applying the permutation function π, where the original sequential ordering of instances is rearranged prior to splitting. As shown, the training and testing sets consist of instances drawn from different positions rather than from contiguous time segments. Specifically, the upper block of the figure represents the training set containing N_{train} randomly selected instances, while the lower block represents the testing set containing N_{test} instances. The ordering within each block reflects the shuffled index sequence $\{\pi(1), \pi(2), \ldots, \pi(N')\}$, rather than the natural chronological order $\{1,2, \ldots, N'\}$. Importantly, although the instances are rearranged across the dataset, the internal structure of each observation remains unchanged. For every selected index $\pi(i)$, the lagged input vector$X_{\pi(i)} = [y_{\pi(i)-1}, y_{\pi(i)-2}, \cdots, y_{\pi(i)-p}]$ remains paired with its corresponding output value $y_{\pi(i)}$. Therefore, the supervised learning relationship between predictors and the target is fully preserved. The random split only alters the global ordering of instances, not the mapping between inputs and outputs.

FFNN and SVR models were trained and evaluated using both random and non-random data split procedures, as described in the preceding sections, to assess their performance under different data partitioning strategies. Random data partitioning was conducted at the level of the constructed supervised learning instances, where each data instance comprised lagged input variables and the corresponding target value, forming a distinct input–output pair. The assignment of instances to training and test sets was performed entirely at random, without retaining sequential ordering or grouping temporally adjacent observations. This design eliminates the risk of optimistic bias arising from partial temporal regime overlap and ensures that performance reflects accurate predictive capability in the random data split

ISSN 2723-6471

procedure. The predictive performance of each fitted model was assessed using error metrics of MAE, MAPE, MSE and RMSE [1], [3], [4], [24].

In the non-random case for the real-world data, the fitted FFNN models were trained 30 times using the same training set. Predictions were generated from each trained model and the average of the forecasted values was computed and compared with the actual test values to calculate the corresponding error metrics. In the random case, the dataset was partitioned into 30 distinct training and testing sets through random sampling. For each partition, the FFNN model was trained 30 times using the same training set to account for variability due to random weight initialization. To ensure the robustness of the fitted model, error metrics were computed by comparing the average forecasted values obtained from 30 runs with their respective test sets. This process was repeated across all 30 random data splits to ensure a comprehensive evaluation of model performance. The resulting error values, reflecting the variability introduced by differing data partitions, are reported as ranges to provide a comprehensive evaluation of model performance.

For SVR models, the training process is deterministic, meaning that with the same data and fixed hyperparameters, the optimization process consistently converges to the same solution. As a result, the model's performance remains consistent across multiple training runs. Therefore, in the non-random case, the error values are reported as they are, while in the random case, the dataset was partitioned into 30 distinct training and testing sets through random sampling. For each partition, the SVR model was fitted using the corresponding training set and predictions were generated and compared with the associated test set to evaluate model performance. The error values, which captured the variability arising from the different data partitions, are presented as ranges to offer a more comprehensive assessment of model performance. Additionally, the accuracy of the forecasts was visually examined through plots comparing actual and predicted values.

4. Results and Discussion

This section presents the combined results and discussion of analyses conducted on simulated and real-world datasets, examining the effect of data split procedures on the predictive performance of FFNN and SVR models.

4.1. Analysis with Simulated Data

Figure 4 presents a representative simulated series from each of the SAR, BL1, BL2, NAR, NMA, TAR, STAR1 and STAR2 models. For every model, an additional 29 replications were produced, resulting in 30 series per specification. To enhance model complexity, Poisson jump processes with jump magnitudes (s_p) of 0.1, 1 and 10 were incorporated. Correspondingly, 30 replications were generated from each extended model. In the simulated data analysis, the primary experimental factor was the data split procedure, while all other hyperparameters remained constant [4], [8]. The FFNN models were trained with a single hidden layer containing one neuron. The Tanh function was used as the activation function for both hidden and output layers. The models were trained using a learning rate of 0.01, over 100 epochs, with a batch size of 16 and a dropout regularization factor set to 0.0001. For the SVR model, the hyperparameters were configured with an epsilon value (ε) of 0.1, a cost parameter (C) of 2 and a gamma (γ) value of 0.25.

Table 1 and table 2 in the Appendix present the forecasting performance of FFNN and SVR models, respectively, under both random and non-random data partitioning across different stationary models. As outlined in the Data Split Procedure section, the error values are expressed as ranges to account for the variability arising from generating each simulated series 30 times with different initial conditions, leading to distinct training and testing sets. Consequently, the final error values for the FFNN and SVR models are reported as ranges, as shown in table 1 and table 2 in the Appendix, capturing fluctuations attributable to both the data generation process and the chosen partitioning strategy. The results reveal distinct differences in performance depending on the chosen splitting approach. For all fitted models, random partitioning produced slightly lower error ranges across all evaluation metrics compared to their non-random split, in both FFNN and SVR models. Incorporating Poisson jumps generally increased error magnitudes; however, random splitting consistently yielded narrower error intervals. Overall, the results demonstrate that random data splitting consistently enhances the forecasting performance and stability of FFNN and SVR models applied to stationary simulated series. This improvement arises from the ability of random partitioning to reduce sensitivity to specific data sequences, thereby strengthening robustness.

ISSN 2723-6471

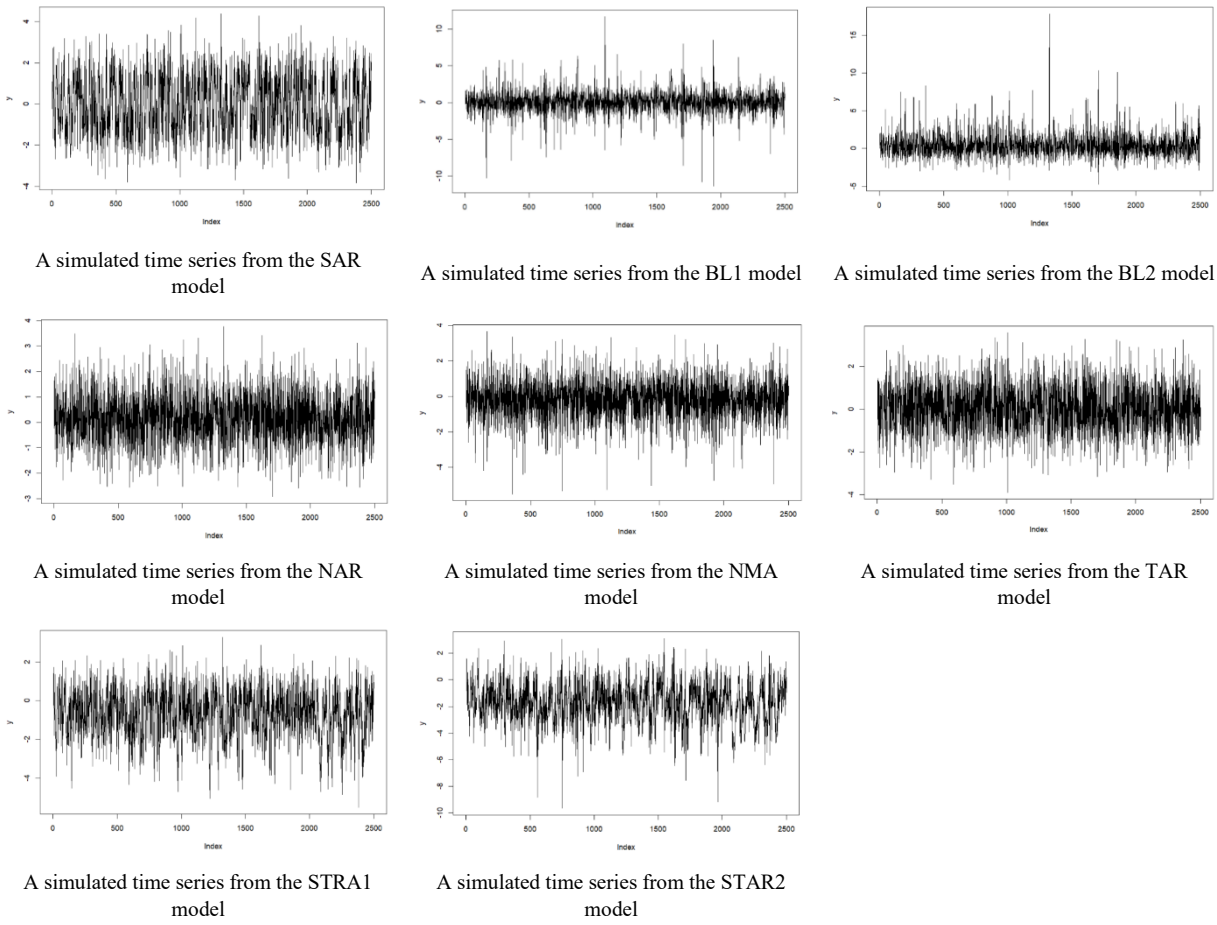

Figure 4. Simulated time series generated from stationary models

The results presented in table 1 summarize the forecasting performance of the FFNN and SVR models under both random and non-random data splitting strategies for simulated non-stationary datasets with σ^2 of 0.1, 1 and 10. The error ranges reveal notable differences in the predictive performance of FFNN and SVR models across random and non-random data splitting strategies for simulated non-stationary datasets with varying noise levels. For the FFNN and SVR models, minimum error ranges were consistently observed under the random data splitting approach in comparison to the non-random method. This indicates that random splitting enables the FFNN and SVR to better capture diverse data patterns, leading to improved forecasting performance and greater adaptability across different test sets. Conversely, the non-random split produced higher error ranges, suggesting potentially less flexible model performances. At the highest noise level ($\sigma^2 = 10$), both FFNN and SVR models showed increased error ranges under both random and non-random data splitting scenarios, reflecting the challenges posed by greater data variability.

Table 1. Forecasting Performance of FFNN and SVR Models using Random and Non-random Data Splitting in Non-stationary Simulated Data

Simulated Data	**FFNN-Random**		**FFNN-Non-Random**	
	MAPE	**RMSE**	**MAPE**	**RMSE**
Simulated series generated with $\sigma^2 = 0.1$	[0.0113,0.0393]	[2.2381,27.9003]	[0.0347,0.0909]	[29.2693,35.1660]
Simulated series generated with $\sigma^2 = 1$	[0.0145,0.0421]	[2.8756,28.2496]	[0.0359,0.0432]	[29.3621,30.4951]
Simulated series generated with $\sigma^2 = 10$	[0.0552,0.0944]	[11.6118,26.7795]	[0.0606,0.0959]	[31.9472,36.5605]
Simulated Data	**SVR-Random**		**SVR-Non-Random**	
	MAPE	**RMSE**	**MAPE**	**RMSE**
Simulated series generated with $\sigma^2 = 0.1$	[0.0275,0.0490]	[3.5326,35.2488]	[0.2126,3.3576]	[30.1748,30.7302]
Simulated series generated with $\sigma^2 = 1$	[0.0276,0.0447]	[3.1285,36.5936]	[0.5568,3.9806]	[28.9215,32.1663]

ISSN 2723-6471

Simulated series generated with $\sigma^2 = 10$	[0.0555,0.0959]	[12.0570,42.6672]	[0.1971,9.1062]	[22.5294,43.3984]

Nonetheless, the FFNN and SVR maintained a relatively lower error under the random data split compared to the non-random split method. Similar patterns were observed in the error ranges of MAE and MSE, reinforcing these findings.

The actual and forecasted graphs corresponding to a selected seed value are presented in figure 1 in the Appendix, where similar results were observed across other seed values. In the graphical representation, the actual values are indicated by dots connected with solid lines, whereas the predicted values are represented by triangle markers linked with dashed lines. It was observed that the actual behavior of the simulated data was well captured in the random data split scenario, with a closer overlap between the actual and forecasted values for both FFNN and SVR models, compared to the non-random split procedure. At the higher noise level of 10, although the forecast values did not closely overlap with the actual observations in either scenario, the general pattern and trend of the data were still reasonably captured by the predicted values in the random split. Overall, these results highlighted that random data partitioning tends to improve the flexibility and predictive performance of both FFNN and SVR models by exposing them to more varied training and testing conditions. Although the non-stationary simulated datasets serve as an illustrative framework incorporating trend and seasonal components, the methodology and conclusions were additionally validated on multiple real-world datasets, ensuring that the observed benefits of random partitioning are not only based on the simulation design but hold under practical forecasting scenarios. This integrated analysis supports the broader applicability of the results and indicates that the observed improvements in model performance arise from the data partitioning strategy itself, rather than from specific characteristics of the simulated data.

4.2. Analysis with Real-world Data

This section presents the analysis results from applying both random and non-random data splitting strategies to real-world datasets using FFNN and SVR models. The selected datasets span a broad range of temporal resolutions, from annual to hourly observations, with sample sizes ranging between 62 and 35,064. They include diverse statistical features, such as long-term trends, seasonal cycles, volatility and abrupt shifts. This heterogeneity provides a robust foundation for evaluating forecasting performance under different data partitioning strategies, allowing for a balanced examination of the strengths and limitations of the applied models.

The analysis primarily aimed to evaluate how different data partitioning strategies affect forecasting performance in real-world time series datasets. To isolate the effect of the data partitioning strategy on forecasting performance, hyperparameter values for FFNN and SVR were initially kept fixed, following the approach used in the simulation study. Simultaneous tuning of hyperparameters alongside different data partition strategies could affect the results, as forecasting performance would then reflect the combined effects of parameter optimization and data splitting. Hence, to ensure consistency and facilitate an unbiased comparison, the hyperparameters of the FFNN and SVR models were maintained at fixed values, consistent with the default configurations provided by the respective R packages. For the FFNN models, the architecture included a single hidden layer comprising five neurons, utilizing the Softsign activation function in both hidden and output layers. The training process employed a learning rate of 0.01, 100 epochs, a batch size of 16 and a dropout rate of 0.0001. In the case of the SVR models, parameter settings included an epsilon value of 0.1, a cost parameter of 1, a gamma parameter of 0.25 and the use of a radial basis function kernel.

The forecasting performance of both FFNN and SVR models under random and non-random data partitioning strategies, based on fixed model architectures, is summarized in table 2. The observed range of errors in the FFNN and SVR models reflects the variability arising from 30 independent random initializations conducted during random data partitioning, which produced multiple distinct training and testing subsets. Accordingly, the reported error values are presented as ranges to capture the influence of random data splitting. In contrast, under non-random partitioning, the SVR model yields single-point error estimates rather than ranges, as training with a fixed partition produces deterministic outcomes without variability across repeated runs. For the non-random split configuration, the FFNN model was trained over 30 iterations and the mean of the resulting forecasted outputs was used for evaluation against the test set. As a result, the corresponding FFNN errors are reported as single averaged values, consistent with the deterministic nature of the fixed partition. Overall, the error ranges for random splits represent variability across multiple train–test partitions. In contrast, single values for non-random splits reflect evaluation on a fixed test set, where this distinction highlights differences in experimental design.

The FFNN model exhibited stable and consistent forecasting performance under random partitioning, with narrow error intervals observed across various data frequencies, ranging from annual to hourly, indicating strong adaptability to

 ISSN 2723-6471

different data structures. Similarly, the SVR model demonstrated notable improvement under random partitioning, particularly for highly volatile datasets such as daily USD/LKR exchange rates and hourly electricity prices, where random splitting more effectively captured complex temporal dependencies. Conversely, models trained under non-random partitions displayed higher and more variable error magnitudes, highlighting weaker generalization due to limited exposure to the full behavior of temporal variation. The monthly cinnamon import dataset, in particular, produced the highest MAPE under the non-random SVR model, suggesting that rigid sequential partitioning can hinder model performance when the data exhibit irregular fluctuations. Overall, the findings highlight that random data partitioning substantially improves the learning and generalization capacity of both FFNN and SVR models, enabling more accurate representation of underlying temporal patterns and fluctuations compared with the non-random approach.

Table 2. Comparison of Forecasting Performance between Random and Non-random Data Partitioning Strategies for the FFNN and SVR Models with Fixed Architectures

Dataset	FFNN Model				SVR Model			
	Non-random		Random		Non-random		Random	
	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	RMSE
Annual average mean surface air temperature	0.0721	0.8187	[0.0198, 0.0626]	[0.2341, 0.644]	0.1887	1.8768	[0.1941, 0.2633]	[0.2931, 0.8215]
Annual life expectancy at birth	0.0401	2.9786	[0.0055, 0.0264]	[0.3643, 2.0822]	0.0197	1.6630	[0.0009, 0.011]	[0.0592, 0.9284]
Quarterly wheat price	0.0559	0.0121	[0.0486, 0.4244]	[0.0108, 0.1412]	0.1659	1.8238	[0.0817, 0.1991]	[0.0194, 0.1214]
Monthly cinnamon import	0.1746	4.0750	[0.1107, 0.2224]	[0.9874, 2.4042]	0.6126	7.2683	[0.1972, 0.2967]	[1.9787, 3.7416]
Monthly crude oil price	0.0810	7.5930	[0.0575, 0.0803]	[4.5969, 7.2338]	0.0683	6.0768	[0.0582, 0.0865]	[2.9581, 6.3946]
Weekly gold price	0.0371	0.0861	[0.0157, 0.0233]	[0.0354, 0.0494]	0.3685	4.1042	[0.0116, 0.0166]	[0.0227, 0.0315]
Daily EURO/LKR	0.0299	7.1819	[0.0068, 0.0088]	[1.8777, 2.2707]	0.1504	32.3911	[0.0093, 0.0267]	[1.1961, 1.5275]
Daily USD/LKR	0.0196	4.0944	[0.0058, 0.008]	[1.1767, 1.6872]	0.0469	10.7325	[0.0047, 0.0058]	[0.9136, 1.1387]
Daily JPY/LKR	0.0132	0.0259	[0.0066, 0.012]	[0.0126, 0.0204]	0.1903	2.0250	[0.0065, 0.0079]	[0.0206, 0.0214]
Hourly electricity price	0.0286	2.6053	[0.017, 0.0465]	[2.8557, 4.5996]	0.0834	3.1552	[0.0413, 0.0915]	[3.1256, 5.1294]

Subsequent analysis involved hyperparameter tuning of both FFNN and SVR models to determine the optimal parameter configurations that maximize forecasting performance. In the initial phase, the comparison between random and non-random data partitioning was conducted using fixed parameter settings to isolate the specific influence of data splitting on model behavior. Thereafter, hyperparameter optimization was implemented to ensure that each model operated under its most effective configuration, thereby enabling a more comprehensive assessment of the relative performance of random and non-random partitioning strategies under optimized learning conditions.

For both random and non-random data splitting strategies, hyperparameter tuning was performed as described in the methodology. The better-performing models were identified based on the minimum observed error values by comparing the actual test and forecast values. Table 3 summarizes the hyperparameter configurations and forecasting performances for FFNN models applied to various real-world datasets under both random and non-random data split procedures. Several key observations were identified from the analysis. Optimal models across various datasets include one to three hidden layers with one to three hidden neurons. However, an increase in the number of hidden layers or neurons beyond four was associated with higher error values. Activation functions commonly used in the two cases of random and non-random split procedures include Tanh, Sigmoid and Softsign, which are frequently applied in the hidden layers and output layer, except for the linear activation function. For both split procedures, when the learning rate is within the range of 0.002 to 0.07, it allows the better-performing models to effectively converge to an optimal solution. The low learning rates ensure gradual updates to the model's weights, preventing drastic changes that could lead to instability and suboptimal convergence. Variations in batch size were observed due to differences in dataset

 ISSN 2723-6471

characteristics, model complexity and optimization stability requirements. In the random split case, a higher number of epochs was observed across all datasets, enabling the model to capture complex patterns through extended training iterations, thereby ensuring the network reaches an optimal solution without premature convergence. Dropout rates remain consistently low in both cases, preventing excessive regularization, allowing the network to retain sufficient information flow while still mitigating overfitting risks.

Table 3. Optimized FFNN Model Parameter Values and Performances for Real-world Datasets under Random and Non-random Data Splitting

Dataset	Data Split	$L_h(N_{hidden})$	AF (Hidden)	AF (Output)	η	N_{epochs}	B_{size}	P_{drop}	MAPE	RMSE
Annual average surface air temperature	Random	1 (2)	Softsign	Sigmoid	0.01	800	16	0.0001	[0.0197,0.0438]	[0.2317,0.5151]
	Non-Random	1 (3)	Tanh	Sigmoid	0.06	500	64	0.0009	0.0686	0.7682
Annual life expectancy at birth data	Random	3 (3, 2, 3)	Tanh, Softsign, Softsign	Sigmoid	0.02	1000	64	0.0001	[0.0021,0.0049]	[0.154,0.5013]
	Non-Random	1 (3)	Softsign	Sigmoid	0.06	1000	128	0.0001	0.0133	1.0552
Quarterly wheat prices	Random	3 (3, 1, 2)	Tanh, Tanh, Tanh	Tanh	0.02	100	16	0.0001	[0.0319,0.3427]	[0.0089,0.1026]
	Non-Random	3 (1, 1, 3)	Tanh, Tanh, Tanh	Tanh	0.01	1000	32	0.0020	0.0329	0.0095
Monthly Cinnamon Import data for the USA	Random	3 (2, 2, 2)	Tanh, Tanh, Tanh	Tanh	0.01	500	32	0.0002	[0.1057,0.1936]	[0.9716,1.9721]
	Non-Random	1 (3)	Tanh	Tanh	0.07	800	32	0.0080	0.1204	3.0817
Monthly crude oil price	Random	3 (3, 3, 2)	Tanh, Softsign, Tanh	Sigmoid	0.009	500	256	0.0007	[0.0483,0.0571]	[3.3777,4.935]
	Non-Random	3 (3, 1, 1)	Tanh, Sigmoid, Tanh	Sigmoid	0.01	600	64	0.0001	0.0590	5.7150
Weekly gold prices	Random	2 (1, 2)	Sigmoid, Softsign	Tanh	0.01	700	64	0.0001	[0.0122,0.0132]	[0.0237,0.029]
	Non-Random	2 (1, 2)	Tanh, Tanh	Tanh	0.01	900	128	0.0004	0.0211	0.0521
Daily EURO/LKR	Random	1 (2)	Sigmoid	Tanh	0.05	1000	16	0.0003	[0.0036,0.0044]	[0.8621,1.0426]
	Non-Random	1 (3)	Softsign	Sigmoid	0.04	100	256	0.0001	0.0197	4.8599
Daily USD/LKR	Random	1 (3)	Sigmoid	Softsign	0.005	100	16	0.0008	[0.0035,0.005]	[0.8208,1.0296]
	Non-Random	1 (2)	Sigmoid	Softsign	0.002	900	16	0.0008	0.0052	1.2499
Daily JPY/LKR	Random	1 (2)	Sigmoid	Softsign	0.008	900	32	0.0001	[0.0039,0.0051]	[0.0081,0.0109]
	Non-Random	1 (2)	Sigmoid	Softsign	0.002	900	16	0.0009	0.0050	0.0109
Hourly electricity price	Random	2 (3, 3)	Sigmoid, Sigmoid	Sigmoid	0.009	300	64	0.0001	[0.0147,0.0365]	[2.1258,2.7384]
	Non-Random	3 (3, 3, 1)	Sigmoid, Tanh, Sigmoid	Sigmoid	0.01	100	16	0.0001	0.0050	2.2902

Table 4 presents the SVR results across various datasets, highlighting variations in the hyperparameters ε, C and γ between the random and non-random data split procedures, along with the corresponding error metrics. In the case of random data partitioning, forecasting errors are reported as ranges because each of the 30 splits produces distinct training and test sets, leading to variability in model performance across splits. In contrast, for the non-random partition, the test set remains fixed and the SVR performance reflects a single evaluation using the optimized hyperparameters values. For the annual average surface air temperature dataset, very low epsilon values were observed for both random and non-random cases, with values as small as 0.00007 and 0.00001, respectively. For the remaining datasets, comparatively higher epsilon values were observed, irrespective of the data partitioning strategy. The cost parameter values observed for the random split procedure were either 1.99, 2, or 0.25, whereas in the non-random split scenario, the values ranged from 0.29 to 4. The gamma parameter remained consistently around 0.25 across most datasets, despite the split procedure, likely because this value provides a balanced trade-off between capturing local patterns and

 ISSN 2723-6471

maintaining generalization. Further, regardless of whether a random or non-random data split procedure was applied, the use of kernel functions other than the radial kernel, such as polynomial, Sigmoid, or linear kernels, led to higher error values.

The error values presented in table 3 and table 4 provide a comparative evaluation of the performance of the FFNN and SVR models under non-random and random data split procedures across various datasets. In general, FFNN and SVR models exhibited lower MAPE and RMSE value ranges for the random split compared to the non-random split, suggesting improved generalization under randomized conditions. A similar behavior was observed for MSE and MAE values.

Table 4. Optimized SVR Model Parameter Values and Performances for Real-world Datasets under Random and Non-random Data Splitting

Dataset	Data Split	ε	C	γ	MAPE	RMSE
Annual average surface air temperature	Random	0.00007	0.25	0.25	[0.027,0.072]	[0.2825,0.7115]
	Non-Random	0.00001	1.21	0.25	0.1165	1.1089
Annual life expectancy at birth data	Random	0.03	2	0.25	[0.0008,0.0029]	[0.0575,0.7041]
	Non-Random	0.2	1.87	0.25	0.0186	1.3909
Quarterly wheat prices	Random	0.1	2	0.25	[0.0521,0.1985]	[0.0099,0.0567]
	Non-Random	0.4	0.29	0.25	0.133	0.0302
Monthly cinnamon import	Random	0.1	0.25	0.25	[0.1399,0.227]	[1.2325,3.1192]
	Non-Random	0.2	1.95	0.01	0.5374	6.7476
Monthly crude oil price	Random	0.03	2	0.25	[0.04,0.0711]	[2.8966,5.3712]
	Non-Random	0.2	4	0.0781	0.053	5.2941
Weekly gold prices	Random	0.02	2	0.25	[0.0112,0.0145]	[0.0219,0.0301]
	Non-Random	0.006	1	0.25	0.0734	0.1943
Daily EURO/LKR	Random	0.02	2	0.25	[0.0091,0.0029]	[1.088,0.7623]
	Non-Random	0.0002	1.98	0.25	0.1234	27.716
Daily USD/LKR	Random	0.01	1.99	0.25	[0.0031,0.0031]	[0.8369,0.8369]
	Non-Random	0.1	1.99	0.25	0.0223	6.1486
Daily JPY/LKR	Random	0.1	2	0.25	[0.0056,0.007]	[0.0106,0.0146]
	Non-Random	0.004	2	0.25	0.0482	0.1069
Hourly electricity price	Random	0.1	2	0.25	[0.0346,0.0373]	2.5572,2.8084]
	Non-Random	0.1	0.65	0.25	0.0431	2.9432

Notably, for the FFNN models, non-random error values fell within the corresponding random error ranges in only three out of ten datasets, namely, the quarterly wheat price, daily JPY/LKR and hourly electricity price datasets. This indicates that for the remaining datasets, the non-random split approach produced higher error values that exceeded the error ranges associated with the random split. A similar trend was observed for the SVR models, where only the quarterly wheat price and monthly crude oil datasets had non-random error values within the random split error ranges, while all other datasets recorded higher error values under the non-random split procedure. Overall, FFNN demonstrated superior performance compared to SVR models under the random data split approach across multiple datasets, including annual average surface air temperature, quarterly wheat price, monthly cinnamon import, weekly gold price, daily JPY/LKR, daily EURO/LKR and hourly electricity price. The datasets of annual life expectancy at birth, monthly crude oil price, weekly gold price and daily USD/LKR demonstrated better performance with the SVR model under the random data split procedure. Nevertheless, across all datasets, spanning various time frequencies including annual, quarterly, monthly, weekly, daily and hourly observations, the lowest error values were consistently achieved under the random data split approach. This consistent performance across different time series data highlights

the robustness of the random sampling strategy in enhancing generalization in predictive modeling. Moreover, by maintaining fixed hyperparameters, it was possible to clearly evaluate the impact of the data split method, which revealed that random partitioning outperformed non-random splitting. Subsequent hyperparameter optimization further improved forecasting accuracy, with random data splitting continuing to yield superior performance relative to the fixed-hyperparameter setting. This two-step approach ensures that the influence of data partitioning is assessed independently before considering the additional benefits of parameter tuning.

Figure 2 in the Appendix displays the actual and fitted graphs for both the random data split scenario using one of the selected seed values and the non-random split case, serving as a representative example of the model's performance under both data partitioning approaches for real-world data. In this representation, actual values are depicted using dots connected by solid lines, while predicted values are illustrated with triangle symbols connected with dashed lines. The actual and fitted graphs exhibited a similar pattern, demonstrating that the random data split procedure effectively captured data movements more accurately than the non-random split for both FFNN and SVR models. A similar pattern was observed across all other seed values, demonstrating that models utilizing the random data split procedure effectively captured the movements of their respective test sets.

Non-random split-based models rarely captured the underlying patterns in the data, with exceptions such as quarterly wheat price and monthly crude oil price. These findings suggest that when the test set lacks large fluctuations, the non-random split technique in both FFNN and SVR models can effectively capture data behavior. However, in the presence of large fluctuations in the test set, this approach fails to accurately represent the data dynamics. In this scenario, the actual and fitted graphs indicated that the predicted values are frequently lower than the actual test values. This occurs because the non-random data split approach trains the model on a fixed subset of data, limiting its exposure to diverse patterns and extreme variations, resulting in an inadequate generalization to large fluctuations in the test set. However, the random data split procedure in FFNN and SVR models effectively captured nearly all the movements in the test set data, including larger fluctuations for all the datasets. This effectiveness can be attributed to the variability introduced through the random data split procedure, which ensures that the training process is exposed to a diverse range of patterns, including fluctuations present in the test set. As a result, the models trained with the random data split procedure develop a more generalized understanding of the data dynamics, leading to improved predictive performance across different datasets.

According to the MAPE-based classification system for forecasting performance, values below 0.1 indicate highly accurate forecasts, those between 0.1 and 0.2 reflect good accuracy, values from 0.2 to 0.5 correspond to reasonable accuracy and values exceeding 0.5 denote poor accuracy [25]. Based on this framework, all real-world datasets fall within the highly accurate category under both random and non-random splitting in FFNN models, except the monthly cinnamon import series. For this dataset, model forecasts correspond to the good accuracy range under both partitioning methods, while the SVR model under non-random splitting records poor performance with a MAPE value above 0.5. Furthermore, under the SVR model with non-random partitioning, the annual mean surface air temperature, quarterly wheat price and daily EURO/LKR datasets are classified within the good accuracy category, whereas the same datasets fall into the highly accurate forecasting group when the random split approach is applied.

Overall, the findings demonstrate that random data partitioning consistently yields narrower error ranges across all datasets, whereas non-random splitting is associated with comparatively higher error levels in certain cases. These results highlight the effectiveness of random data partitioning in enhancing the robustness and predictive reliability of FFNN and SVR models when applied to real-world time series, compared to the non-random splitting approach. The findings of this will benefit researchers and practitioners in the fields of time series forecasting and machine learning by providing empirical insights into data partitioning strategies, thereby supporting more informed model selection and evaluation practices. Although MAPE is widely used due to its intuitive percentage interpretation, it may become unstable when actual observations approach zero and overly sensitive in highly volatile series. To mitigate this limitation, forecasting performance in this study was evaluated using multiple complementary error measures, including MSE, MAE and RMSE. In addition, future research may consider alternative scale-independent measures such as the symmetric mean absolute percentage error or the mean absolute scaled error, which provide more robust evaluation in the presence of near-zero values or strong volatility. Another key limitation of the present study lies in the focus on a restricted set of neural network configurations.

Although the analysis provides valuable insights into the influence of network complexity and data partitioning strategies, the findings may not fully generalize across alternative architectures. Future research could extend this study by exploring a wider range of other appropriate machine learning models and comparative analyses incorporating

 ISSN 2723-6471

ensemble learning approaches or hybrid frameworks, which may also provide further improvements in forecasting accuracy and robustness. Additionally, time-series-aware resampling techniques such as rolling-origin evaluation and blocked cross-validation could be employed to systematically compare their performance with the random and non-random data partitioning strategies considered in this study.

5. Conclusion

Data partitioning refers to the process of dividing a dataset into distinct subsets for model development and evaluation. This step plays a critical role in statistical modeling and machine learning, as it directly influences the reliability and generalizability of predictive outcomes. Proper partitioning ensures that models are trained on one portion of the data and tested on another, thereby reducing the risk of overfitting and enabling a more accurate assessment of model performance. Several partitioning strategies are commonly applied, each with its own advantages. Non-random approach preserves inherent temporal or structural characteristics of the dataset, making them particularly relevant for time series applications. Random partitioning distributes data points into training and testing subsets without regard to order, promoting balanced representation of patterns and variability. This approach helps assess the robustness of the model, as it prevents the model from learning specific temporal dependencies that could limit its generalization ability.

Hence, this study systematically evaluated the performance of FFNN and SVR models in time series forecasting, comparing the forecasting effectiveness of random and non-random data split procedures through an extensive experimental analysis utilizing both simulated and real-world datasets spanning multiple temporal frequencies, including hourly, daily, weekly, monthly, quarterly and annual time intervals. The results revealed that the random data split procedure consistently demonstrated superior capability in capturing the underlying dynamics of the test sets for both FFNN and SVR models, surpassing the performance of the non-random split approach. This enhanced performance can be attributed to the variability introduced by the random partitioning, which facilitated the development of diverse training and testing sets. Such diversity enabled the models to learn more representative patterns and generalize more effectively to unseen data, thereby improving predictive accuracy. Analysis of both stationary and non-stationary simulated models demonstrated that the FFNN and SVR models consistently achieved lower error ranges under the random data splitting procedure compared to the non-random approach. This advantage of random partitioning remained evident even as model complexity increased. Across all real-world datasets, the lowest error values were consistently observed when the random data split approach was employed, both under fixed parameter settings and following hyperparameter optimization. In contrast, non-random split-based models struggled to capture the underlying data patterns, with exceptions observed in datasets such as quarterly wheat prices and monthly crude oil prices. These results suggested that when the test set lacks significant fluctuations, the non-random split technique in both FFNN and SVR models can still effectively capture the data's behavior.

However, when large fluctuations are present in the test set, this approach fails to accurately reflect the data dynamics. This limitation arises because the non-random data split procedure trains the model on a fixed data subset, restricting its exposure to diverse patterns and extreme variations, which impairs its ability to generalize to large fluctuations. Conversely, the random data split procedure in FFNN and SVR models effectively captured various fluctuations in the data, which can be attributed to the variability introduced by the random partitioning, allowing the models to learn from a broader range of data patterns. From a practical perspective, these findings provide clear guidance for practitioners and researchers in time series forecasting, emphasizing that careful consideration of data partitioning is essential for reliable model evaluation. For instance, financial analysts forecasting exchange rates, energy companies predicting demand and meteorologists conducting weather forecasts can benefit from adopting random data-splitting strategies in combination with appropriate models. Compared to non-random splits, these approaches reduce bias, enhance evaluation accuracy and result in more robust and dependable predictions, as demonstrated in the present study. By comparing random and non-random data partitioning strategies using both simulated and real-world datasets, this research provides new insights, demonstrating that the random data split procedure enhances predictive performance compared to the non-random approach for time series predictions.

6. Declarations

6.1. Author Contributions

Conceptualization: B.R.P.M.B. and N.V.C.; Methodology: B.R.P.M.B. and N.V.C.; Software: B.R.P.M.B.; Validation: B.R.P.M.B. and N.V.C.; Formal Analysis: B.R.P.M.B. and N.V.C.; Investigation: B.R.P.M.B.; Resources: N.V.C.; Data Curation: B.R.P.M.B.; Writing Original Draft Preparation: B.R.P.M.B.; Writing Review and Editing: B.R.P.M.B. and N.V.C.; Visualization: B.R.P.M.B.; All authors have read and agreed to the published version of the manuscript.

6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6.3. Funding

This study is funded by the University Research Grant awarded by the University of Peradeniya (Grant No: URG/2024/49/S).

6.4. Institutional Review Board Statement

Not applicable.

6.5. Informed Consent Statement

Not applicable.

6.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] R. Adhikari and R. K. Agrawal, "Forecasting strong seasonal time series with artificial neural networks," *J. Sci. Ind. Res. (India).*, vol. 71, no. 10, pp. 657–666, 2012.

[2] R. S. Al-Gounmeein and M. T. Ismail, "Forecasting the exchange rate of the Jordanian dinar versus the US dollar using a Box-Jenkins seasonal ARIMA model," *Int. J. Math. Comput. Sci.*, vol. 15, no. 1, pp. 27–40, 2020.

[3] T. Mong and U. Ngan, "Forecasting foreign exchange rate by using ARIMA model: a case of VND/USD exchange rate," *Res. J. Finance Account.*, vol. 7, no. 12, pp. 38–44, 2016.

[4] G. P. Zhang, B. E. Patuwo, and M. Y. Hu, "A simulation study of artificial neural networks for nonlinear time-series forecasting," *Comput. Oper. Res*., vol. 28, no. 4, pp. 381–396, 2001, doi: 10.1016/S0305-0548(99)00123-9.

[5] B. R. P. M. Basnayake and N. V. Chandrasekara, "Utilization of various optimization algorithms in neural network models for forecasting exchange rates," *Int. J. Adv. Comput. Res*., vol. 2024, no. Jan., pp. 55–60, 2024, doi: 10.1109/ICARC61713.2024.10499767.

[6] T. Fischer, C. Krauss, and A. Treichel, "Machine learning for time series forecasting: a simulation study," *arXiv,* vol. 2018, no. Jan., pp. 1–15, 2018.

[7] G. P. Zhang and M. Qi, "Neural network forecasting for seasonal and trend time series," *Eur. J. Oper. Res*., vol. 160, no. 2, pp. 501–514, 2005, doi: 10.1016/j.ejor.2003.08.037.

[8] B. R. P. M. Basnayake and N. V. Chandrasekara, "Assessing the performance of feedforward neural network models with random data split for time series data: a simulation study," *Int. J. Smart Comput. Syst*., vol. 2024, no. Jan., pp. 1–6, 2024, doi: 10.1109/SCSE61872.2024.10550735.

[9] H. Ince and T. B. Trafalis, "A hybrid model for exchange rate prediction," *Decis. Support Syst*., vol. 42, no. 2, pp. 1054–1062, 2006, doi: 10.1016/j.dss.2005.09.001.

[10] I. S. Al-Mejibli, J. K. Alwan, and D. H. Abd, "The effect of gamma value on support vector machine performance with different kernels," *Int. J. Electr. Comput. Eng*., vol. 10, no. 5, pp. 5497–5506, 2020, doi: 10.11591/IJECE.V10I5.PP5497-5506.

[11] K. Smets, B. Verdonk, and E. M. Jordaan, "Evaluation of performance measures for SVR hyperparameter selection," *IEEE Access*, vol. 2007, no. Jan., pp. 637–642, 2007, doi: 10.1109/IJCNN.2007.4371031.

[12] A. J. Dautel, W. K. Härdle, S. Lessmann, and H. V. Seow, "Forex exchange rate forecasting using deep recurrent neural networks," *Digit. Finance*, vol. 2, no. 1–2, pp. 69–96, 2020, doi: 10.1007/s42521-020-00019-x.

[13] H. Tong and K. Lim, "Threshold autoregression, limit cycles and cyclical data," *J. R. Stat. Soc*., vol. 42, no. 3, pp. 245–268, 1980, doi: 10.1111/j.2517-6161.1980.tb01126.x.

[14] C. Granger and A. Anderson, An introduction to bilinear time series models. Göttingen: Vandenhoeck and Ruprecht, 1978.

[15] T. Terasvirta and H. M. Anderson, "Characterizing nonlinearities in business cycles using smooth transition autoregressive models," *J. Appl. Econometrics*, vol. 7, no. S1, pp. S119–S136, 1992, doi: 10.1002/jae.3950070509.

[16] B. R. P. M. Basnayake and N. V. Chandrasekara, "Forecasting exchange rates in Sri Lanka: a comparison of the double seasonal autoregressive integrated moving average models (DSARIMA) and SARIMA models," *J. Sci. Univ. Kelaniya,* vol. 15, no. 2, pp. 192–209, 2022, doi: 10.4038/josuk.v15i2.8067.

[17] N. V. Chandrasekara and C. D. Tilakaratne, "Forecasting exchange rates using artificial neural networks," *Sri Lankan J. Appl. Stat*., vol. 10, no. Jan., pp. 187–201, 2009.

[18] A. Emam and H. Min, "The artificial neural network for forecasting foreign exchange rates," *Int. J. Serv. Oper. Manag.*, vol. 5, no. 6, pp. 740–757, 2009, doi: 10.1504/IJSOM.2009.026772.

[19] M. S. Islam and E. Hossain, "Foreign exchange currency rate prediction using a GRU-LSTM hybrid network," *Soft Comput. Lett.*, vol. 3, no. Dec., pp. 1–10, 2021, doi: 10.1016/j.socl.2020.100009.

[20] V. Pacelli, V. Bevilacqua, and M. Azzollini, "An artificial neural network model to forecast exchange rates," J. *Intell. Learn. Syst. Appl.*, vol. 3, no. 2, pp. 57–69, 2011, doi: 10.4236/jilsa.2011.32008.

[21] V. Plakandaras, T. Papadimitriou, and P. Gogas, "Forecasting daily and monthly exchange rates with machine learning techniques," *J. Forecast.,* vol. 34, no. 7, pp. 560–573, 2015, doi: 10.1002/for.2354.

[22] W. Zhao, T. Tao, and E. Zio, "Parameter tuning in support vector regression for reliability forecasting," *Chem. Eng. Trans.*, vol. 33, no. Jan., pp. 523–528, 2013, doi: 10.3303/CET1333088.

[23] W. J. Chen, J. J. Yao, and Y. H. Shao, "Volatility forecasting using deep neural network with time-series feature embedding," *Econ. Res.-Ekon. Istraz*., vol. 36, no. 1, pp. 1377–1401, 2023, doi: 10.1080/1331677X.2022.2089192.

[24] G. Weisang and Y. Awazu, "Vagaries of the euro: an introduction to ARIMA modeling," *Case Stud. Bus. Ind. Gov. Stat.,* vol. 2, no. 1, pp. 45–55, 2008.

[25] C. D. Lewis, Industrial and business forecasting methods: A practical guide to exponential smoothing and curve fitting. London: Butterworth Scientific, 1982.