P-ISSN: 2618-0723
E-ISSN: 2618-0731

NAAS Rating: 5.04
www.extensionjournal.com

International Journal of Agriculture Extension and Social Development

Volume 7; Issue 9; September 2024; Page No. 192-198

Received: 14-06-2024
Accepted: 12-07-2024

Indexed Journal
Peer Reviewed Journal

Comparative analysis of time series forecasting: Traditional Statistical Approaches vs. Machine Learning Methods

[1,2]Prabhat Kumar, [1,2]Satyam Verma, [1,2]Manoj Varma, [1,2]Kaushal Kumar Yadav and [1,2]Ankit Kumar Singh

[1]The Graduate School, ICAR-Indian Agricultural Research Institute, New Delhi, Delhi, India

[2]ICAR-Indian Agricultural Statistics Research Institute, New Delhi, Delhi, India

DOI: https://doi.org/10.33545/26180723.2024.v7.i9c.1054

Corresponding Author: Satyam Verma

Abstract

In the evaluation of time series predictors, the model selection processes for traditional forecasting methods and machine learning (ML) techniques differ significantly. This study aims to assess their performance across multiple forecasting horizons using an extensive set of univariate time series. By comparing the out-of-sample accuracy of popular ML methods with traditional statistical approaches, the results indicate that traditional methods consistently outperform ML methods across all accuracy measures and forecasting horizons considered. However, ML techniques can be improved through preprocessing methods such as Box-Cox and log transformation. The results show that log transformation enhances the performance of ML techniques compared to traditional statistical forecasting methods. ML techniques must improve in accuracy, efficiency, and interpretability to be competitive. The primary contribution of this research is demonstrating the superior accuracy of traditional statistical methods over ML methods and highlighting the urgent need to identify the underlying causes and develop strategies to improve ML performance.

Keywords: Forecasting, machine learning, price series, statistical method

1. Introduction

Time series forecasting techniques have become essential tools across a wide range of applications, including the analysis and prediction of technical, physical, and economic data. They are crucial for making significant decisions with far-reaching impacts, necessitating a comprehensive evaluation of their overall effectiveness. Accurate error estimation of predictors is vital not only as an indicator of system reliability and accuracy but also for selecting the best forecasting method and parameter configuration.

Many researchers have proposed various methods to measure the accuracy of forecast algorithms for univariate time series data. However, these methods often have limitations, lack precise definitions, or are unbounded, potentially leading to incorrect conclusions. Significant research has been conducted on the potential of artificial intelligence (AI), specifically machine learning (ML) methods and neural networks (NNs), to enhance time series forecasts. AI has made considerable contributions to forecasting, with numerous publications introducing new ML algorithms, methodologies, and accuracy improvements.

The primary objective of both ML and statistical techniques is to minimize a loss function, typically the sum of squared errors, to improve forecasting accuracy. They differ in their approaches: statistical methods use linear processes, while ML methods employ non-linear algorithms. ML techniques are computationally more complex and depend heavily on computer science, situating them at the intersection of statistics and computer science. It is crucial to objectively assess the overall performance of ML approaches in forecasting, as this has not been thoroughly done, raising concerns about their effectiveness in advancing forecasting research and accuracy.

Numerous studies have employed NN techniques, comparing their accuracy with traditional statistical methods. Neural networks were first used for forecasting in 1964, but significant progress was made only after the introduction of the backpropagation algorithm about 20 years later (Zhao, 2009) [23]. Ahmed *et al*. (2010) [2] reviewed numerous studies, some dating back to 1995, and noted that the results were mixed. Similarly, Adya and Collopy (1998) [1] analyzed 48 NN experiments and found inconsistent results compared to statistical approaches.

The first large-scale study using 3003 time series was the M3 Competition, published in 2000. Makridakis (2000) [14] utilized an Automated Artificial Neural Network (AANN) method, which performed below the most accurate statistical methods in the competition. Eleven years later, Crone *et al*. (2011) [6] published the results of a specialized NN competition using a subset of the M3 monthly data, comparing 22 NN and computational intelligence (CI) methods alongside 11 statistical methods. The primary goal of this research is to empirically determine whether ML techniques outperform statistical methods and how their advantages can be leveraged to improve forecasting

accuracy. Previous studies concluded that no ML approach surpassed the Theta method. This research aims to establish empirically the conditions under which ML methods may be superior to statistical approaches and how their benefits can be utilized to enhance forecasting accuracy.

2. Statistical and ML methods

To evaluate the performance of machine learning (ML) algorithms against classical statistical approaches, we considered two accurate statistical methods: Simple Exponential Smoothing (SES) and Autoregressive Integrated Moving Average (ARIMA). SES is designed to forecast series without a trend, while ARIMA is employed to understand the data and predict future trends. In contrast, the machine learning approaches utilized in this evaluation include the Multi-Layer Perceptron (MLP) and Bayesian Neural Network (BNN).

2.1 Statistical Methods

2.1.1 Simple Exponential Smoothing (SES)

Exponential smoothing is a time series forecasting method originally designed for univariate data but can be adapted to handle data with a systematic trend or seasonal component. As a powerful alternative to the well-known Box-Jenkins ARIMA family of models, this technique produces forecasts by calculating weighted averages of past observations, with weights that decrease exponentially over time. Essentially, more recent observations carry more weight.

$$F_{t+1} = \alpha Y_t + (1 - \alpha) F_t \quad (1)$$

The forecast for the next period (F_{t+1}) is determined by assigning a weight (α) to the most recent observation (Y_t) and a weight of 1 − α to the most recent forecast (F_t). A smaller value of α (e.g., 0.1) results in greater smoothing, while a larger value of α (e.g., 0.9) provides less smoothing. Alternatively, one can select α from a range of values (e.g., α = 0.1, 0.2, …, 0.9) and choose the one that minimizes the Mean Squared Error (MSE).

2.1.2 Autoregressive Integrated Moving Average (ARIMA)

The ARIMA model is one of the most significant and commonly utilized models in time series analysis. Its widespread use is largely due to its robust statistical characteristics and the well-established Box–Jenkins methodology for model construction. The ARIMA (p, d, q) model, where p, *d*, and *q* represent the orders of the autoregressive, differencing, and moving average components, respectively can be formulated as follows:

$$\varphi(B)\Delta^d x_t = \theta(B) u_t \quad (2)$$

The backshift operator *B* is defined as $Bx_t = x_{t-1}$, where, x_t represents the value of the price series at time *t*. The differencing operator Δ is expressed as $(1 - B)$. The polynomials in φ(*B*) and θ(*B*) have degrees *p* and *q*, respectively. Additionally, the disturbance term at time *t*, denoted as u_t, is a random variable characterized by a mean of zero and a constant variance of σ^2.

2.2 Machine Learning Methods

2.2.1 Multi-Layer Perceptron (MLP)

The multilayer perceptron (MLP), as shown in Figure 1, is a network of interconnected neurons or nodes that enables nonlinear mapping between an input layer and an output layer within a model. The connections between these nodes are represented by weights, and the output signals are generated through simple nonlinear activation functions based on the sum of the node's inputs. These weights scale the output from one node and pass it as input to nodes in the next layer of the network. This directional flow of information is what characterizes the multilayer perceptron as a "feed-forward neural network." In the MLP structure, the input layer receives the inputs, and the output layer produces the outputs. Typically, an MLP consists of an output layer and at least one hidden layer. Each node in the MLP is fully connected to the nodes in the adjacent layers. Research has shown that with appropriate adjustments to the weights and activation functions, a multilayer perceptron can approximate any smooth, measurable function between the input and output vectors (Hornik *et al*., 1989) [10]. The MLP is a widely used neural network architecture for both classification and regression tasks in modern applications. The formulation of the MLP is as follows:

$$\hat{y} = v_0 + \sum_{j=1}^{NH} v_j g\left(w_j^T \grave{x}\right) \quad (3)$$

In this context, w_j represents the weight vector associated with the j^{th} hidden node, while $v_0, v_1, \ldots, v_{NH}$ denote the weights for the output node. The network's output is denoted by $\hat{y}$ .The input vector $\grave{x}$ is derived by augmenting the original input vector *x* with a constant term, such that $\grave{x} = (1, x^T)^T$. The function *g* captures the output of the hidden nodes, typically defined by a squashing function. The Multi-Layer Perceptron (MLP) is highly parameterized, with its complexity influenced by the choice of the number of hidden nodes *NH* (Van Dijk *et al*., 2002) [21].

2.2.2 Bayesian Neural Network (BNN)

A Bayesian neural network (BNN) is a neural network constructed according to a Bayesian probabilistic formulation (MacKay, 1992), thereby linking BNNs to the Bayesian parameter estimation principle in classical statistics and the regularization concept found in methodologies such as ridge regression. The methodology of BNN shares similarities with the Multilayer Perceptron (MLP) approach, as it optimizes network parameters using Bayesian principles, wherein weights are estimated under predefined error distributions. BNN finds extensive applications across various domains, including engineering, finance, and economics. Its fundamental objective lies in treating network parameters or weights as random variables, governed by predefined distributions that promote models of moderate complexity or those generating smooth fits. Posterior distributions of weights are evaluated after data observation, enabling computation of network predictions. These predictions account for both the smoothness enforced

by priors and the accuracy of fit conferred by observed data. The regularization aspect of BNN involves formulating and minimizing the objective function.

$$J = ED + (1 - a)EW \quad (4)$$

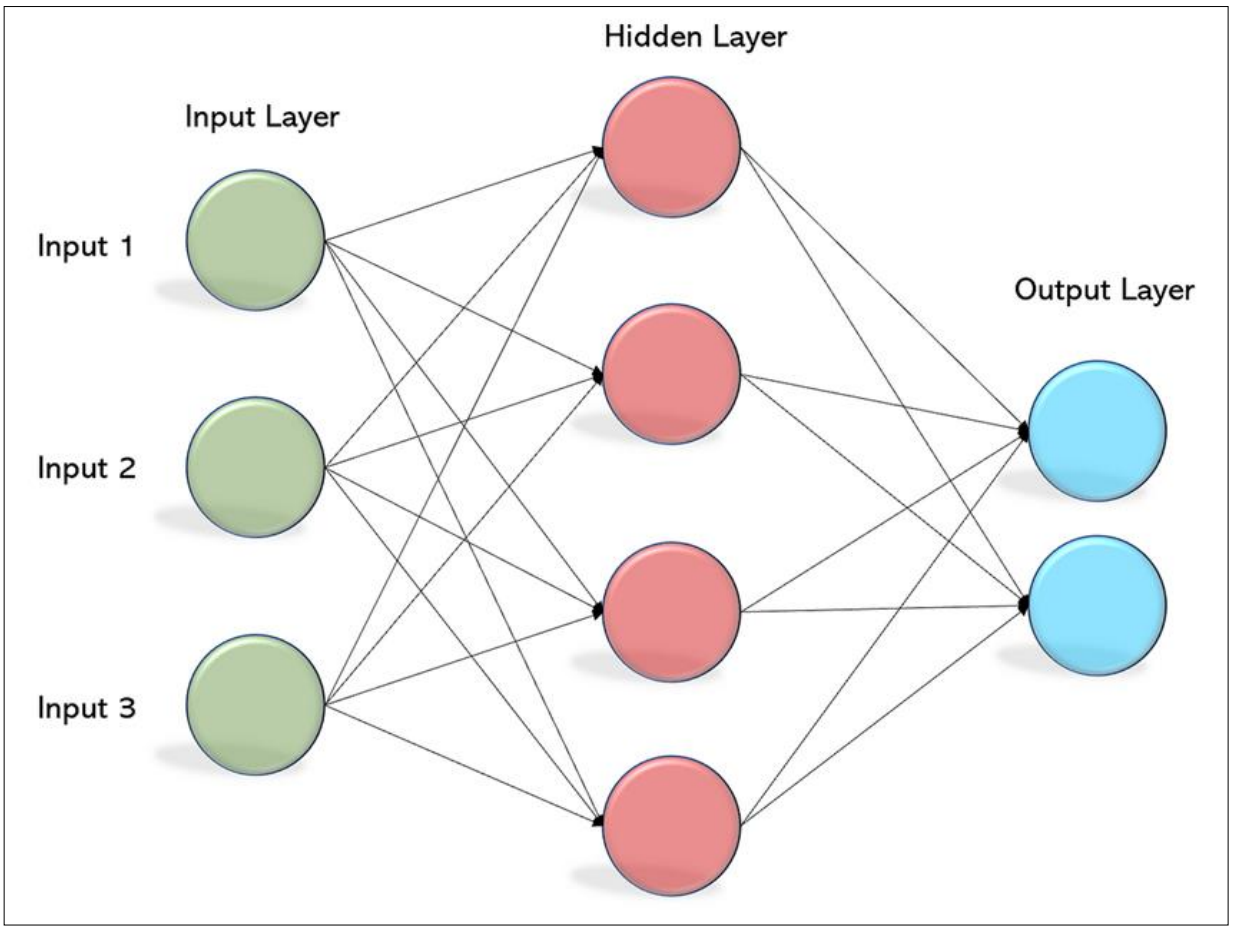

Fig 1: Multilayer perceptron with one hidden layer

In the given formulation, ED represents the summation of the square errors in the network outputs, EW denotes the summation of the squares of the network parameters, specifically the weights, and α stands for the regularization parameter.

In the Bayesian technique, a common practice for selecting the prior involves utilizing a normal density distribution. This choice assigns augmented significance to smaller network parameter values.

$$p(w) = \left(\frac{1-\propto}{\pi}\right)^{\frac{L}{2}} e^{-(1-\alpha)} E_W \quad (5)$$

Where L denotes the number of parameters (weights). The posterior is then given by

$$P(w/D, \propto) = \frac{p(D/w, \propto)\, p(w/\propto)}{p(D/\propto)} \quad (6)$$

Let D denote the gathered dataset. The probability density of the dataset given the parameters can be computed as follows, assuming normally distributed errors:

$$P\ (D/w, \propto) = \left(\frac{\propto}{\pi}\right)^{\frac{M}{2}} e^{-(\alpha)} E_W \quad (7)$$

By replacing the density expressions in both the prior and the data's probability density function into the posterior, where M denotes the number of training data points, we can gain a deeper insight into the relationship between these variables.

$$P\ (w/D, a) = cexp(-J) \quad (8)$$

Where c is normalizing constant. The regularization constant is also determined using Bayesian principals, from

$$P(a/D) = \frac{p(D/a)\, p(a)}{p(D)} \quad (9)$$

To achieve optimal weights and parameters, it is essential to maximize p(w/D, a) and p(a/D) correspondingly. The expression for p(D/a) involves a quadratic approximation of J concerning the weights, followed by the integration of these weights.

3. Data Pre-processing

A data mining approach known as data pre-processing involves transforming raw data into a more comprehensible format. Real-world data often exhibits inadequacies, inconsistencies, and deficiencies in specific patterns or trends, leading to numerous inaccuracies. In the evolution of time series collection techniques, achieving stationarity in both mean and variance is deemed crucial. Within the realm of machine learning (ML), differing perspectives exist regarding the necessity of pre-processing. Some studies assert that ML methods are inherently capable of effectively modeling various data patterns and can thus be applied directly to the original data. Conversely, other research contends that without adequate pre-processing, ML algorithms may exhibit instability and yield subpar results. The effectiveness of subsequent forecasting endeavors can be greatly influenced by pre-processing the time series data. Modifying the time series data through pre-processing techniques simplifies the task of prediction modeling. Three primary types of pre-processing methods include trend removal, log or power transformations, and seasonal adjustments. MLP (Multi-Layer Perceptron) models are often unable to accurately capture seasonality, despite claims to the contrary. The activation function of MLP models tends to exhibit instability, which can be mitigated by detrending the series, thereby achieving greater consensus regarding the trend.

3.1 Original data: No pre-processing is applied.

3.2 Transforming the data

3.2.1 Log Transformation

Log transformation is applied to reduce the variability in the data. Among various transformation techniques employed to mitigate skewness in data and approximate it to normality, log transformation stands out as a prominent method. When the original data conforms to a log-normal distribution or closely resembles one, the application of log transformation results in a dataset that conforms to a normal or nearly normal distribution.

3.2.2 Box-Cox Transformation

In the context where *t* represents the time period and λ denotes a parameter, with *w* signifying the modified variable and *y* indicating the target variable, it is noted that when the parameter value equals one, the dataset inherently exhibits a normal distribution, thereby obviating the necessity Box-Cox transformation.

$$w_t = \begin{cases} \frac{y_t^{\lambda}-1}{\lambda}, & \lambda > 0 \\ log\, y_t, & otherwise \end{cases} \quad (10)$$

4. Data description

The study utilized the monthly wholesale price data of potatoes in the Delhi market as a case study, sourced from the Ministry of Consumer Affairs webpage for India, spanning from January 2010 to July 2020. The original pricing series was partitioned into two subsets: a training set consisting of 115 observations, and a testing set comprising data from the preceding year for validation purposes. Descriptive statistics of the three series employed in the analysis are provided in Table 1, while Figure 2 depicts the time plots of the respective series.

5. Accuracy measures

Two measures of accuracy are used first is SMAPE and MASE

5.1 Symmetric Mean Absolute Percentage Error (SMAPE)

$$\text{sMAPE} = \frac{2}{K}\sum_{t=1}^{k}\frac{|Y_t - \hat{Y}_t|}{|Y_t|-|\hat{Y}_t|} \times 100$$

In this scientific framework, let *k* indicate the forecasting horizon, Y_t denote the actual observations, and $\hat{Y}_t$ represent the forecast produced by the model at time *t*. It is important to note that the symmetric Mean Absolute Percentage Error (sMAPE) imposes heavier penalties for large positive errors than for negative ones.

5.2 Mean Absolute Scaled Error (MASE)

$$\text{MASE} = \frac{\frac{1}{k}\sum_{t=1}^{k}|Y_t - \hat{Y}_t|}{\frac{1}{n-m}\sum_{t=m+1}^{n}|Y_t - Y_{t-m}|}$$

In time series analysis, where *n* denotes the number of historical observations and *m* indicates the frequency of the time series, it is important to note that the Mean Absolute Scaled Error (MASE) is characterized by its scale independence, among other properties.

Table 1: Descriptive statistics of potato wholesale price (Rs./q) of Delhi markets

Statistics	Delhi
Observations	127.00
Mean	967.00
Median	875.30
Maximum	2464.40
Minimum	372.80
Standard deviation	447.45

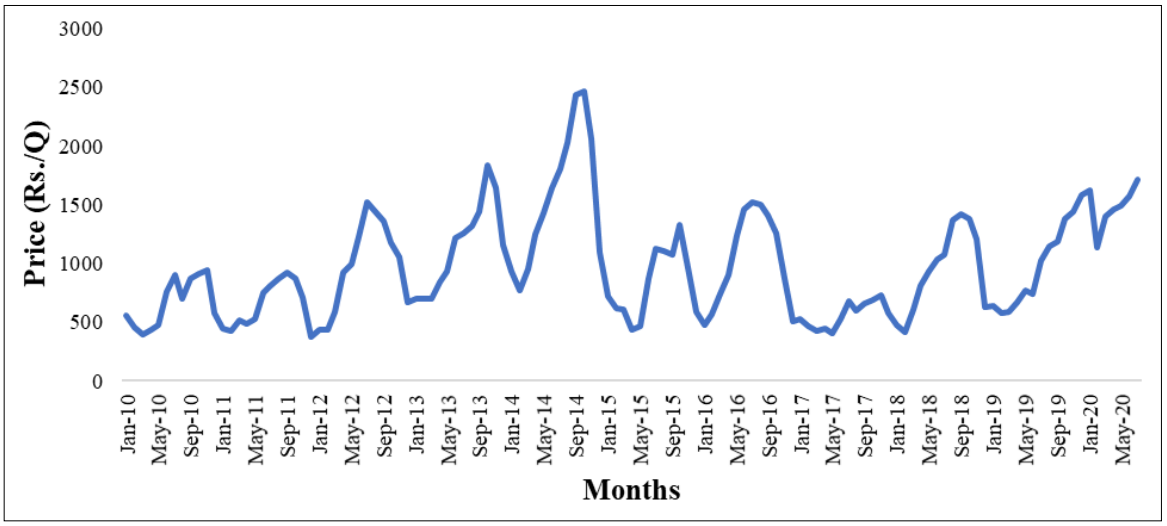

Fig 2: Time plot for monthly Delhi potato wholesale price

Table 2: Forecasts value of the ML, Statistical methods, and most appropriate pre- processing alternative of Potato wholesale price (Rs/q) of Delhi market

Months	Actual	MLP	BNN	ARIMA	SES	MLP (LOG)	MLP (Box Cox)
1	1142.50	1088.53	1125.67	1025.27	1123.35	1123.76	1359.25
2	1183.33	1153.31	1120.36	1189.36	1150.77	1156.30	1369.57
3	1384.48	1235.58	1208.95	1027.87	1370.14	1402.69	1759.89
4	1436.66	1322.17	1456.23	1365.98	1398.45	1426.32	1529.39
5	1587.90	1388.77	1480.34	1378.63	1423.26	1563.35	1578.81
6	1625.00	1459.25	1521.36	1581.24	1489.52	1569.38	1602.54
7	1137.50	1524.17	1254.17	1529.31	1527.69	1094.28	986.57
8	1400.00	1588.94	1324.10	1289.78	1000.34	1406.98	1356.94
9	1460.83	1652.92	1478.30	1378.29	1489.24	1450.58	1487.67
10	1494.35	1710.69	1523.69	1402.33	1469.52	1502.69	1547.62
11	1569.16	1765.77	1625.37	1489.56	1496.59	1609.35	1802.69
12	1711.11	1820.83	1689.71	1669.45	1693.36	1823.9	1968.34

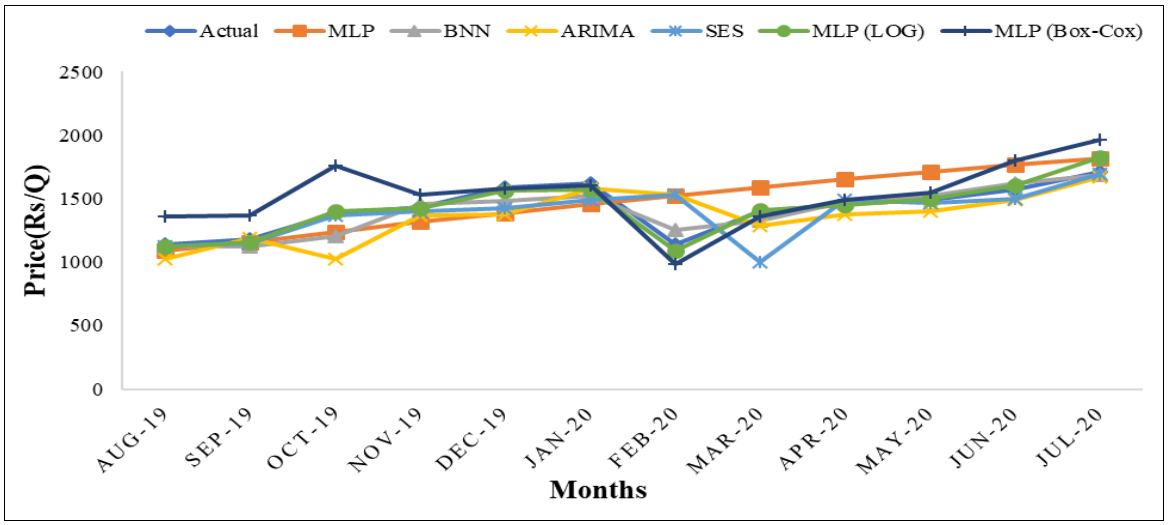

Fig 3: Plots of 12 months forecasted values of different models along with actual Delhi market

Table 3: Forecasting performance of the ML, Statistical methods, and most appropriate pre-processing alternative

Models	sMAPE(%)	MASE
MLP	24.33	0.98
BNN	10.77	0.94
ARIMA	21.65	0.83
SES	19.99	0.86
MLP(LOG)	5.87	0.85
MLP(BOX-COX)	24.53	0.82

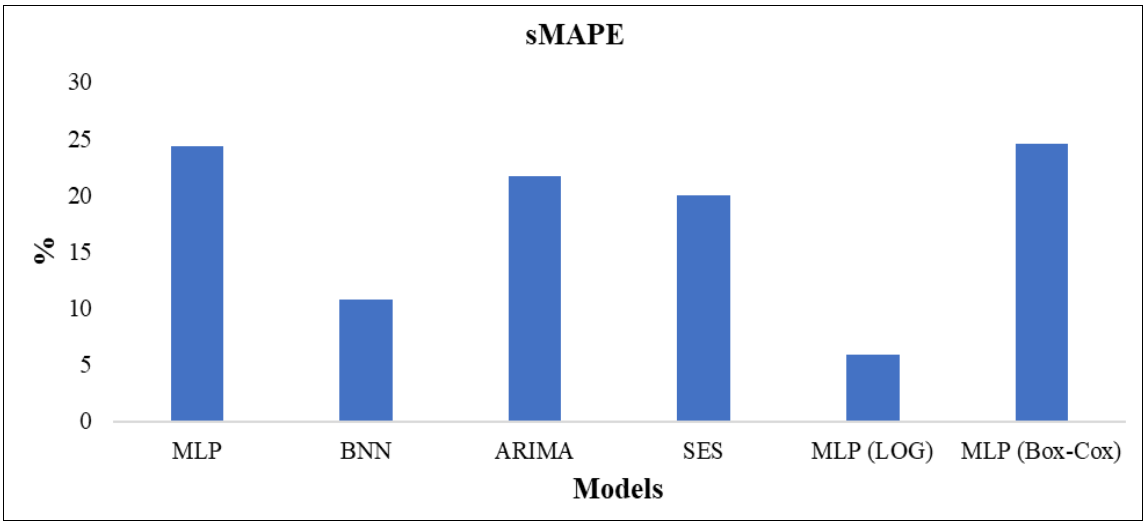

Fig 4: Forecasting performance of the ML, Statistical methods, and most appropriate pre-processing alternative

6. Results and Discussion

Today's forecasting heavily relies on machine learning techniques, which are expected to yield higher overall accuracy. Table 2 presents 12-month estimates generated using a range of time series models, showcasing the forecasting performance of these models and benchmark models against two evaluation criteria for a 12-month forecast horizon, as detailed in Table 3. Figure 4 illustrates that statistical methods outperform ML approaches. Specifically, MLP achieves a 2.68% lower SMAPE value compared to ARIMA, which is somewhat unexpected. On the other hand, BNN performs favorably in comparison to statistical methods.

To enhance the accuracy of MLP, the most effective approach is preprocessing the data, which involves ensuring data stability and reducing noise points. While there are several preprocessing techniques available, we have applied Log and Box-Cox transformations to the same dataset, resulting in improved sMAPE results when compared to preprocessing alone. However, this transformation does not lead to an increase in MASE accuracy for MLP methods.

7. Conclusions

While it may be disappointing that machine learning models exhibit lower forecasting accuracy compared to statistical methods, researchers hold a strong optimism regarding their vast potential in forecasting applications. Clearly, there is a need for additional efforts to develop these systems, a common requirement for new techniques, including complex forecasting methods that have significantly improved accuracy over time. This study's key contribution is highlighting that traditional statistical techniques outperform machine learning techniques, underscoring the urgency of identifying underlying causes and developing solutions. Data pre-processing plays a critical role in enhancing the performance of ML algorithms, and for better results, it is preferable to employ log pre-processing methods rather than ML methods, simplifying the data.

At this juncture, several empirically verifiable suggestions/speculations can be put forth as potential ways forward for ML methods:

- Pre-processing data before applying ML techniques, leading to simpler models, reduced computation requirements, and faster learning.
- Grouping series into homogeneous categories and constructing ML algorithms optimized for each.
- Guarding against over-fitting, as it remains uncertain whether ML models can effectively distinguish noise from data patterns.
- Incorporating uncertainty estimation in point forecasts and providing data for creating confidence intervals around these projections.

8. Competing interests

The authors have no potential conflicts to report that are important to the article's content.

9. References

1. Adya M, Collopy F. How effective are neural networks at forecasting and prediction? A review and evaluation. J Forecast. 1998;17(5-6):481-495.
2. Ahmed NK, Atiya AF, Gayar NE, El-Shishiny H. An empirical comparison of machine learning models for time series forecasting. Economet Rev. 2010;29(5-6):594-621.
3. Assimakopoulos V, Nikolopoulos K. The theta model: A decomposition approach to forecasting. Int J Forecast. 2000;16(4):521-530.
4. Bergmeir C, Benítez JM. On the use of cross-validation for time series predictor evaluation. Inform Sci. 2012;191:192-213.
5. Box GE, Jenkins GM, Reinsel GC, Ljung GM. Time series analysis: Forecasting and control. John Wiley & Sons; c2015.
6. Crone SF, Hibon M, Nikolopoulos K. Advances in forecasting with neural networks? Empirical evidence from the NN3 competition on time series prediction. Int J Forecast. 2011;27(3):635-660.
7. Gardner Jr ES. Exponential smoothing: The state of the art-Part II. Int J Forecast. 2006;22(4):637-666.
8. Goodwin P, Lawton R. On the asymmetry of the symmetric MAPE. Int J Forecast. 1999;15(4):405-408.
9. Green KC, Armstrong JS. Simple versus complex forecasting: The evidence. J Bus Res. 2015;68(8):1678-1685.

10. Hornik K, Stinchombe M, White H. Multi-layer feedforward networks are universal approximators. Neural Networks. 1989;2:359-366.
11. Hyndman RJ, Koehler AB. Another look at measures of forecast accuracy. Int J Forecast. 2006;22(4):679-688.
12. MacKay DJ. Bayesian interpolation. Neural Comput. 1992;4(3):415-447.
13. Ismail MM, Nicholson A, Abu-Mostafa Y. Financial markets, very noisy information processing. Proc IEEE. 1998;86(11):2184-2195.
14. Makridakis S, Hibon M. The M3-Competition: results, conclusions, and implications. Int J Forecast. 2000;16(4):451-476.
15. Makridakis S, Wheelwright SC, Hyndman RJ. Forecasting methods and applications. John Wiley & Sons; c2008.
16. Makridakis S. The forthcoming Artificial Intelligence (AI) revolution: Its impact on society and firms. Futures. 2017;90:46-60.
17. Makridakis S, Spiliotis E, Assimakopoulos V. Statistical and Machine Learning forecasting methods: Concerns and ways forward. PLOS One. 2018;13(3).
18. Leshno M, Lin VY, Pinkus A, Schocken S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks. 1993;6(6):861-867.
19. Lippmann R. An introduction to computing with neural nets. IEEE ASSP Mag. 1987;4(2):4-22.
20. Sharda R, Patil RB. Connectionist approach to time series prediction: An empirical test. J Intell Manuf. 1992;3(5):317-323.
21. Dijk VD, Terasvirta T, Franses PH. Smooth transition autoregressive models: A survey of recent developments. Econom Rev. 2002;21:1-47.
22. Zhang GP, Qi M. Neural network forecasting for seasonal and trend time series. Eur J Oper Res. 2005;160(2):501-514.
23. Zhao L. Neural networks in business time series forecasting: Benefits and problems. Rev Bus Inf Syst. 2009;13(3).