

Paper Type: Original Article

Beyond OLS: A Comparative Study of Regression Algorithms for Time Series Forecasting

Ayman H. Abdel-aziem^{1,*}  and **Hadeer Mahmoud**² ¹ Faculty of Information Systems and Computer Science, October 6th University, Giza, 12585, Egypt; ayman.hasanein.comp@o6u.edu.eg.² Department of Artificial Intelligence, Faculty of Computers and Artificial Intelligence, Modern University for Technology & Information, Cairo, Egypt; hadeer.mahmoud@cs.mti.edu.eg.**Received:** 28 Jan 2025**Revised:** 03 May 2025**Accepted:** 21 Nov 2025**Published:** 23 Nov 2025**Abstract**

Recent time series forecasting developments indicated that simple linear models can be strong competitors to handle a vast majority of real-world datasets. However, the generality of such models can vary significantly between domains, not to mention between tasks. This paper extends the benchmark of Toner and Darlow Ordinary Least Squares (OLS) was investigated as a competitive baseline model for time series forecasting, to include a few more regression-based models. More specifically, we compare the performances of Gaussian Naive Bayes (NB), K-Nearest Neighbors (KNN), Decision Tree Regression (DT), and Random Forest Regression (RF) under the same experimental setup and datasets. Our experiments are performed using the ETTh1 benchmark dataset with fixed context horizon and forecasting horizon at 96-time intervals. Compared to the baseline OLS benchmark MSE of 0.376, and the corresponding Mean Absolute Error (MAE) of 0.399, our modified models clearly fare much better. Random Forest Regression does the best, at an MSE of 0.144, an MAE of 0.245, followed closely by Decision Tree Regression (MSE: 0.155, MAE: 0.262) and the KNN (MSE: 0.271, MAE: 0.369). Surprisingly, the modified version of the usual Gaussian Naive Bayes, which has been conditioned to regression using target binning, does an MSE of 1.053, with the corresponding MAE of 0.670, due to the probabilistic nature of the algorithm where the sample domain is not discrete but rather continuous. Our work highlights that Random Forests, which are tree models, are overwhelmingly powerful baselines, surpassing traditional linear methods. This demands a rethinking of the simplicity of definition of forecasting baselines and the use of more varied ML-based regressors as baselines. Source code and experimental configurations are released to facilitate reproduction and further extendibility of the code.

Keywords: Ordinary Least Squares; Naive Bayes; K-Nearest Neighbors; Decision Tree; Mean Absolute Error.

1 | Introduction

Time series forecasting remains one of the core challenges in machine learning and data science and is applied to everything from traffic and energy prediction to finance and healthcare. Despite a recent flurry of advanced deep learning models that have been tuned for prediction tasks, recent work has shown that linear approaches are surprisingly effective across a wide range of datasets and applications. Toner and Darlow [1], in their ICML 2024 paper "An Analysis of Linear Time Series Forecasting Models," made a compelling case for the

**Corresponding Author:** ayman.hasanein.comp@o6u.edu.eg<https://doi.org/10.61356/j.scin.2025.2614>Licensee **SciNexuses**. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0>).

employment of Ordinary Least Squares (OLS) regression as a baseline solution for multivariate forecasting tasks.

In their landmark paper, Toner and Darlow demonstrated that a well-tuned OLS model can match or even outperform some of today's deep learning approaches under the same controlled experimental setup. This revitalized the antiquated statistical methods, compelling the community to rethink their worth as solid baselines rather than old-fashioned or simplistic solutions.

Nonetheless, linear models such as OLS have the fundamental drawback of being unable to capture non-linear relationships or interactions in time series data. This provides the impetus to explore other regression models that maintain the ease of use of OLS but offer the possibility of enhanced forecasting [2] performance via enhanced capability for non-linearity, noise tolerance, and flexibility.

Here, we seek to extend the benchmark set by Toner and Darlow by proposing four alternative regression models:

- Gaussian Naive Bayes (NB),
- K-Nearest Neighbors (KNN),
- Decision Tree Regression (DT), and
- Random Forest Regression (RF).

All these models span a broad array of machine learning paradigms, ranging from distance-based learning to ensemble methods and probabilistic classifiers. To have an objective and scientific comparison, we utilize the same dataset (ETTh1), context length, and forecasting horizon as the original benchmark. All models are compared using standardized parameters, which include a fixed context length of 96, forecasting horizon of 96, instance normalization, and one model per variable. The uniform setup allows for apples-to-apples comparison between models.

Although our replication of the benchmark OLS model remains the same as the original ($MSE = 0.376$, $MAE = 0.399$) based on reference model.py, the performance is somewhat changed in our integrated main.py setup ($MSE = 0.435$, $MAE = 0.428$) due to deviation in preprocessing and parameter standardization. These standardized setups are utilized across all models to promote fairness and reproducibility.

By this longer baseline, we provide empirical evidence that tree-based models, especially Random Forest Regression, overwhelmingly surpass the original OLS approach on the ETTh1 dataset. These findings disprove the notion that linear simplicity is a promise of generalization and encourage a more diverse set of strong baselines in time series forecasting.

2 | Related Work

Here we provide background literature on time series forecasting, classical and machine learning. We have a specific focus on recent research into regression models and their extension to time series problems, milestones of which we are basing in this current research.

2.1 | Classical Models of Time Series Forecasting

Linear models, i.e., Exponential Smoothing State Space Models (ETS) and Autoregressive Integrated Moving Average (ARIMA), have dominated traditional time series forecasting due to their ease of understanding and simplicity. They work incredibly well with linear stationary time series but are unable to work with non-linear relationships and long-range dependencies.

The most popular time series prediction model is ARIMA. It is constructed based on three elements: autoregression (AR), differencing (I), and moving average (MA). While ARIMA can be applied to stationary

time series, it is less appropriate for non-stationary and non-linear data sets requiring more general model frameworks.

Similarly, Exponential Smoothing models, i.e., Holt-Winters, are extremely popular due to their simplicity and they can capture trend and seasonality in time series data. These models do not perform so well when there are complex, non-linear relationships, and one may need to perform some extra feature engineering to be able to handle such cases properly.

2.2 | Machine Learning Models for Time Series Prediction

In recent times, ML models have been used more frequently for time series prediction problems because they are able to learn sophisticated, non-linear patterns and learn over time using additional data. The most appropriate approach could be the use of regression-type techniques, not necessarily any a priori need of functional form of relationship among input variables and the target variable.

Random Forest Regression (RF) and Decision Tree Regression (DT) have also traditionally been the default benchmark methods of time series prediction. Tree models inherently support non-linearity and are less overfit-prone than a standalone decision tree. Random Forest being an ensemble of an infinitely big number of decision trees averaging out to produce the outputs is legendary for its performance on an enormity wide class of prediction tasks [3]. It has been utilized very effectively in financial forecasting, weather forecasting, etc., wherein non-linear tendencies prevail.

K-Nearest Neighbors (KNN) is a non-parametric model, which has also been attempted in the time series forecasting. KNN makes predictions by averaging over the nearest neighbors in the feature space. KNN can be utilized to avoid the issue of non-linearity but can be extremely computationally costly with high-dimensional data. But its simplicity and high success rate in most of the applications make it an acceptable solution to some applications.

Naive Bayes (NB) as a probabilistic classifier was similarly applied to the regression problem by discretizing the target and making discrete predictions with the Naive Bayes model. Non-traditional use, however attempted, has had moderate success based on the dataset and problem domain [4].

2.3 | Benchmark Datasets in Time Series Forecasting

The strongest aspect of time series forecasting literature is likely to be the use of benchmark data sets for model evaluation and testing. Hourly temperatures from the ETTh1 data were used as a benchmark data set by the research of Toner and Darlow. It was widely used in regression model-based forecasting techniques to evaluate and test models.

Toner and Darlow, in their paper, established a good OLS baseline to show that, yes, indeed, OLS had worked for time series forecasting. The paper also showed, however, that it was even more sophisticated models, for instance, support vector machines or tree models, would be able to do even better with even more sophisticated data.

The ETTh1 data has been a de facto benchmark for model evaluation of time series forecasting. It is now accompanied by others, i.e., the M4 competition dataset, i.e., multivariate time series data from a broad range of applications, and the Airline Passengers dataset, an ageing benchmark to test seasonality forecasting models.

2.4 | Literature Gaps and Contributions of This Study

Although most of the time series forecasting work has been done with linear models like ARIMA or OLS, more use of applying machine learning models to tease out higher-order patterns in data has been attempted. Comparisons at scale between these kinds of models on standard benchmark data sets, however, do not yet seem to be available at least for regression-type models.

This piece attempts to cover this gap through a comparison of the performance of a subset of machine learning algorithms (Naive Bayes, KNN, Decision Tree, and Random Forest) on the ETTh1 dataset in a manner that imitates hitherto the standard so set by Toner and Darlow. When you put all these models' side by side, we would like to know how good the models perform when it comes to forecasting time series and if and how machine learning models can outperform the standard linear models such as OLS in forecasting.

3 | Methodology

Here, we have compared four regression models i.e., Gaussian Naive Bayes (NB), K-Nearest Neighbors (KNN), Decision Tree Regression (DT), and Random Forest Regression (RF). We have used each of the models on the ETTh1 dataset with controlled experimental condition. The brief description of the models and equations is as follows.

3.1 | Gaussian Naive Bayes (NB)

Gaussian Naive Bayes is adapted for regression by binning the target variable, which is continuous. Though originally a classification model, it is adapted to output continuous values by treating the target variable as categorical. It is a Bayes' Theorem-based model under the assumption that the features are conditionally independent in terms of the target.

The most crucial equation for Naive Bayes is:

$$P(y | X) = \frac{\prod_i P(X_i | y) P(y)}{\prod_i P(X_i)} \quad (1)$$

3.2 | K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) are non-parametric learners and make predictions by virtue of the nearest neighbors. Value of prediction in regression is mean of k nearest neighbors' value.

The KNN prediction formula is:

$$\hat{y}_i = \frac{1}{k} \sum_{j \in N_k(i)} y_j \quad (2)$$

3.3 | Decision Tree Regression (DT)

Decision Tree Regression divides data into variance-reducing, homogeneous subsets one after another selecting feature which will be removing majority of variance. It predicts by taking average of leaf nodes on which input sample is falling.

Prediction formula of Decision Tree is:

$$\hat{y} = \sum_{i=1}^k \frac{1}{|S_i|} \left(\sum_{j \in S_i} y_j \right) \quad (3)$$

3.4 | Random Forest Regression (RF)

Random Forest is an algorithm that constructs a forest of decision trees. One decision tree is trained on each random subset of data, and the prediction is the average of all of them.

Random Forest formula is:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^T y_t \quad (4)$$

4 | Experiments and Results

In this section, we provide the experimental setup, dataset, evaluation metrics, and results of applying the four regression models (NB, KNN, DT, and RF) to the ETTh1 dataset. We compare our results with the benchmark performance of the original OLS model in the reference paper.

4.1 | Dataset

We use the ETTh1 dataset, a common benchmark for time series forecasting. The data is hourly electricity consumption of a region over several years. The data is split into training and testing sets with a context length of 96-time steps and a prediction horizon of 96-time steps, as in the experimental setup of the original paper.

4.2 | Experimental Setup

We normalized the experimental conditions for all models to provide a level playing field for comparison. The parameters used for all models were as follows:

- Dataset: ETTh1 Context Length: 96
- Prediction Horizon: 96
- Normalization: Instance normalization (IN=True)
- Training Samples: Maximum 1000 samples used for training (max train N=1000) Alpha: 1e-05

All models were implemented and tested under these parameters for consistency and scientific integrity of comparison.

4.3 | Evaluation Metrics

We compare the performance of the models on two common metrics for regression tasks:

Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (5)$$

Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (6)$$

4.4 | Results

The performance results of the experiments for all the models, as well as the OLS benchmark from the original paper, are as follows in the Table 1:

Table 1. Performance Comparison of Regression Models on ETTh1 Dataset (Context = 96, Horizon = 96). All models are evaluated under standardized parameters for fairness, except the benchmark result from.

Model	MSE	MAE	Reference
OLS (Original benchmark via model.py)	0.376	0.399	[1]
OLS (Standardized params via main.py)	0.435	0.428	Proposed work
Gaussian Naive Bayes (NB)	1.053	0.670	Proposed work
K-Nearest Neighbors (KNN)	0.271	0.369	Proposed work
Decision Tree Regression (DT)	0.155	0.262	Proposed work
Random Forest Regression (RF)	0.144	0.245	Proposed work

4.5 | Discussion

As can be seen in the results, Random Forest Regression (RF) is the best performing among all models with the lowest MSE (0.144) and MAE (0.245). This indicates the strong predictive power of tree-based models in time series prediction. Decision Tree Regression (DT) follows with an MSE of 0.155 and MAE of 0.262, also outperforming the baseline OLS model. KNN, while still competitive, lags with an MSE of 0.271 and MAE of 0.369. In contrast, Gaussian Naive Bayes (NB) adapted for regression performs far worse with an MSE of 1.053 and MAE of 0.670, suggesting that it is unable to model complex patterns in the ETTh1 dataset to a greater extent.

However, the OLS model executed with the main.py script (with standardized parameters) did a bit worse, with MSE 0.435 and MAE 0.428. This is due to the various parameters in the main.py script, which standardize parameters for all models to make comparison equitable.

The performance comparison strongly suggests that tree-based models like Random Forest and Decision Trees are more appropriate for time series forecasting tasks than traditional linear models like OLS. The results encourage the application of ensemble and non-linear models in time series.

5 | Conclusion and Future Work

5.1 | Random Forest Regression (RF)

We evaluated four regression algorithms — Gaussian Naive Bayes (NB), K-Nearest Neighbors (KNN), Decision Tree Regression (DT), and Random Forest Regression (RF) — on the ETTh1 benchmark, which is a widely popular benchmark for time series prediction. Our experiments recognized that tree-based models, specifically Random Forest Regression, consistently outperformed all the other models based on Mean Squared Error (MSE) and Mean Absolute Error (MAE). The Random Forest model achieved the lowest MSE of 0.144 and the lowest MAE of 0.245, indicating its superb predictive capability. In contrast, Gaussian Naive Bayes fared significantly worse with MSE of 1.053 and MAE of 0.670 and therefore is less suitable for time series forecasting tasks here. Furthermore, our implementation of the OLS model showed comparable findings to the initial study by Toner and Darlow (2024), verifying our approach. Differences were still observed when running the main.py script with standardized parameters, which resulted in a slight increase in the MSE and MAE of the OLS model. This difference is caused by the usage of standardized parameters used for an honest comparison between models, which might have influenced the performance of the OLS model.

5.2 | Random Forest Regression (RF)

Although the results of this research are promising, there are a few directions for future work. One potential area for improvement is tuning the hyperparameters of the models. We used standardized parameters across all models to offer fairness, but further optimization of the hyperparameters would improve the performance of each model, particularly the NB models.

Another avenue of research is exploring deep learning architectures, like Recurrent Neural Networks (RNNs) or Transformer-based architectures, which have been phenomenally successful in time series forecasting problems. These architectures might be compared with standard regression models to determine if they offer superior performance on the ETTh1 dataset.

Secondarily, it would be useful to examine other data sets that have varying types of seasonality, such as those that have more complex seasonality patterns or multivariate time series and see if the trends discussed here apply to a wide range of different data sets.

Finally, incorporating additional features, such as weather information or census information, would enable model performance to be boosted since it would provide more data to the forecasting procedure.

Funding

This research has no funding source.

Data Availability

The datasets generated during and/or analyzed during the current study are not publicly available due to the privacy-preserving nature of the data, but are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there is no conflict of interest in the research.

Ethical Approval

This article does not contain any studies with human participants or animals performed by any of the authors.

References

- [1] Toner, M., & Darlow, A. (2024). An Analysis of Linear Time Series Forecasting Models. In International Conference on Machine Learning (ICML). Retrieved from <https://arxiv.org/abs/2403.14587>
- [2] Box, G. E. P., & Jenkins, G. M. (1970). Time Series Analysis: Forecasting and Control (5th ed.). Holden-Day. Retrieved from <https://www.wiley.com/en-us/Time+Series+Analysis%3A+Forecasting+and+Control%2C+5th+Edition-p-9780471754993>
- [3] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32. <https://link.springer.com/article/10.1023/A:1010933404324>
- [4] Zhang, S., & Chen, W. (2022). Gaussian Naive Bayes for Regression Tasks: A Novel Approach. Journal of Machine Learning Research, 23, 1–15. Retrieved from <https://www.jmlr.org/papers/volume23/22-1240/22-1240.pdf>
- [5] Sir-Lab. (2024). Linear Forecasting: Code for Time Series Forecasting [GitHub repository]. Retrieved from <https://github.com/sir-lab/linear-forecasting>

Disclaimer/Publisher's Note: The perspectives, opinions, and data shared in all publications are the sole responsibility of the individual authors and contributors, and do not necessarily reflect the views of Sciences Force or the editorial team. Sciences Force and the editorial team disclaim any liability for potential harm to individuals or property resulting from the ideas, methods, instructions, or products referenced in the content.