Time Series Forecasting Techniques for Indoor Temperature Prediction

Anwesha Samaddar, Mohit Yadav

***Abstract*—A variety of data science techniques have been utilized for developing time series forecasting models that provide insightful information in various realms like weather and climate, financial markets, energy management, and healthcare. This report presents a comprehensive study of popular machine-learning algorithms for indoor temperature prediction of a room within a solar house. The short-term temperature forecast data would assist in reducing energy consumption in an HVAC (Heating, Ventilation, and Air Conditioning) system.**

I. Introduction

One of the most economically significant applications of Machine learning is time series forecasting and a significant amount of research has been dedicated to building cutting-edge prediction models. On the other hand, forecasting can be highly difficult, for example, due to covariate shift - where data distribution is different in the test set as compared to our training data. We rely on historical data for building forecasting models, therefore changes in the underlying patterns in the future would degrade model accuracy. Continuous monitoring and periodical updates to the model are critical for reducing the impact of covariate shifts.

In this work, we demonstrate multiple approaches and their outcomes of short-term indoor temperature forecasting utilizing data acquired by a complex monitoring system as input. Our system aims to reduce energy consumption related to the Heating, Ventilation, and Air Conditioning (HVAC) units by deciding whether or not to activate the HVAC system based on temperature prediction data. HVAC accounts for 53.9% of total consumption, and the energy necessary to maintain temperature is less than that required to lower/increase it.

For developing such intelligent systems, Artificial Intelligence based methods - specifically Soft computing and Machine learning have been widely used in literature. Artificial Neural Networks (ANN) based predictive models provide high performance particularly while capturing non linear trends in energy systems. Martínez et. al. [1] demonstrated forecasting room air temperature using ANN while taking into account predicted weather factors like solar radiation and air temperature, as well as manipulating actuators like heating, ventilation, and cooling systems as variables.

To develop a robust forecasting model [2] promotes the use of online learning algorithms, emphasizing the significance of building models from scratch or adapting pre-trained models to new settings.

We will now introduce the basic domestic system architecture (of our solar house) and the various controlled variables and sensor measurements that are utilized for forecasting temperature.

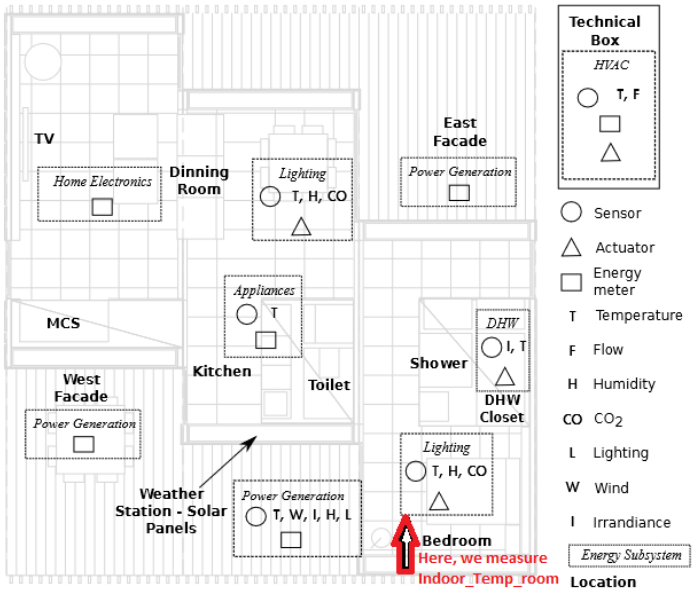

Fig. 1: System sensors and actuators map for Solar powered house. Sourced from [3].

We use the data collected for a modular solar-powered home made mostly of wood (as illustrated in Fig. 1), showing an innovative way to achieve near-zero energy status. The house uses passive techniques and water heating technologies to reduce its dependency on external electrical power. The power source includes solar power generated by photovoltaic panels, a Domestic Hot Water (DHW) system, as well as a grid link for extra electrical power when needed.

HVAC, house appliances, and lighting/home electronics are the three major consumers of energy, with the HVAC system being the largest. The solar system effectively generates electricity and a device manages this energy for optimal consumption within the house or and sends surplus energy to the grid or battery storage.

A software controller monitors the energy generation by integrating multiple sensors to measure voltage, current, power values, and a network of actuators and sensors for temperature readings. Forecasting techniques play a critical role in energy-efficient environments due to their ease of incorporation into predictive control systems.

II. Dataset Desciption

The dataset originates from the monitoring system installed in a solar house constructed by the University of CEU Cardenal Herrera for the Solar Decathlon Europe 2012 competition. Approximately 40 days of monitoring data is present that has been sampled every minute, and smoothened with 15-minute means. This is a multivariate time-series dataset including dates, sensor measurements, weather measurements, and other information.

Following are the covariates in our dataset:

- Id of the measure
- Date in UTC
- Time in UTC
- Carbon dioxide in ppm for dining room
- Carbon dioxide in ppm for room
- Relative humidity (dining room) in %
- Relative humidity (room) in %
- Lighting (dining room) in Lux
- Lighting (room) in Lux
- Rain (proportion of the last 15 minutes where rain was detected, in the range [0,1])
- Sun dusk
- Wind in m/s
- Sun light in west facade in Lux
- Sun light in east facade in Lux
- Sun light in south facade in Lux
- Sun irradiance in W/m^2
- Outdoor relative humidity in %
- Day of the week (computed from the date), 1=Monday, 7=Sunday

Our target variable is **Indoor room temperature** in ºC. Fig.2 shows a visualization of these covariates with respect to time and Fig. 3.

Evaluation metric: We shall use the Mean Squared Error (MSE) for quantifying the differences between predicted values by forecasting models and the corresponding actual observed values. The MSE is calculated as follows:

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

III. Methods and Results

Some traditional statistical methods for time series forecasting include ARIMA and SARIMAX. ARIMA stands for AutoRegressive Integrated Moving Average that combines autoregression (AR), differencing (I), and moving averages (MA) to capture specific features of time series data. SARIMAX which stands for Seasonal AutoRegressive Integrated Moving Average with eXogenous influences, is an ARIMA modification that incorporates exogenous variables, allowing the model to account for external factors impacting the time series. Recurrent neural networks (RNNs) with LSTMs is another powerful method for capturing long-term dependencies

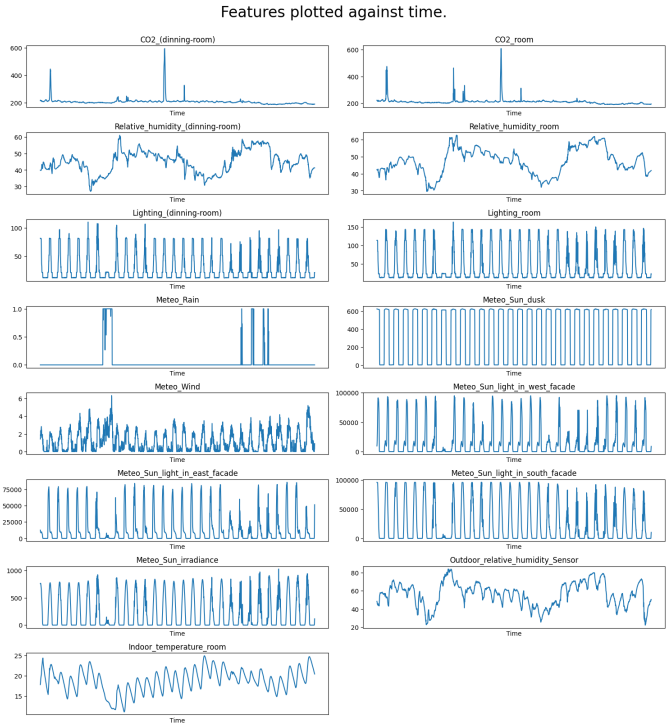

Fig. 2: Visualization of the weather covariates with respect to time present in our dataset.

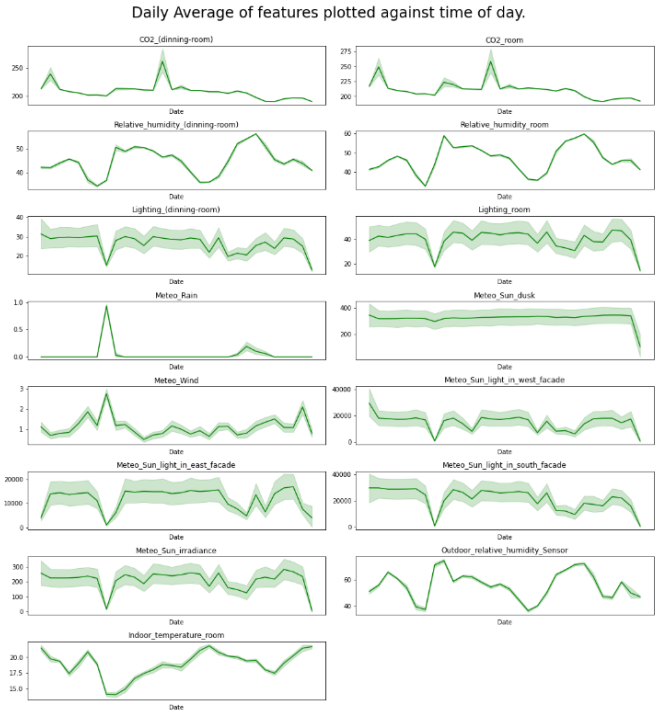

Fig. 3: Visualization of the daily averages of weather covariates in our dataset.

in sequential data, which makes it effective for time series prediction. For this project we have looked at four methods to

predict the temperature for two in advance.

Tree-based estimators like Decision Trees and its extensions like Random Forests or XGBoost are unable to extrapolate data and hence face challenges while dealing with covariate shift.

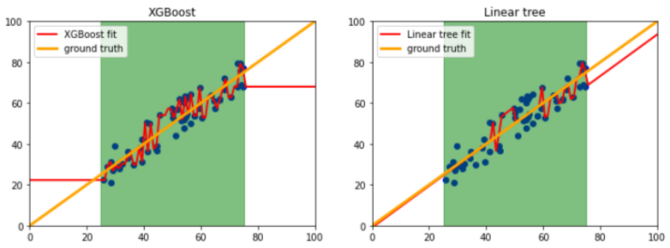

Fig. 4: An illustration highlighting XGBoost's inability to extrapolate on random iid samples drawn from the same distribution. Sourced from [4].

A. *Linear Regression model trained with all weather features*

Here we assume indoor room temperature depends on all the weather covariates provided i.e information related to CO_2 measurements, relative humidity, lighting, rain, sun dusk, wind, sunlight in lux, solar irradiance, and outdoor relative humidity. We add features for dayofyear, time and a sequential Id for the model to learn from these. Then we try fitting a linear regression model will these columns as features and indoor room temperature as our target variable.

LinearRegression function from the 'sklearn' python package has been used to fit all the weather-dependent features. We get an MSE of 2.684 on the training data and an MSE of 1.661 on the test set. Fig. 5 and Fig. 6 show performances of our LR model on train and test data respectively.

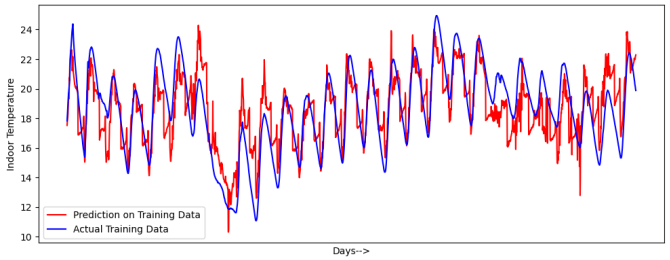

Fig. 5: Visualizing temperature forecasting results of LR model (trained on weather and date/time features) on the **training set**.

From the plot we see that Linear Regression model has abruptness in the predictions, however is able to capture the overall trend of the target variable.

B. *Linear Regression with lag features*

In time series forecasting, lag features are used by integrating prior observations of the target variable as input features to predict future values. A lag feature represents the target variable's value at a previous time step. This notion is based on the assumption that past values of the variable can be used to predict future values. As the target variable of temperature

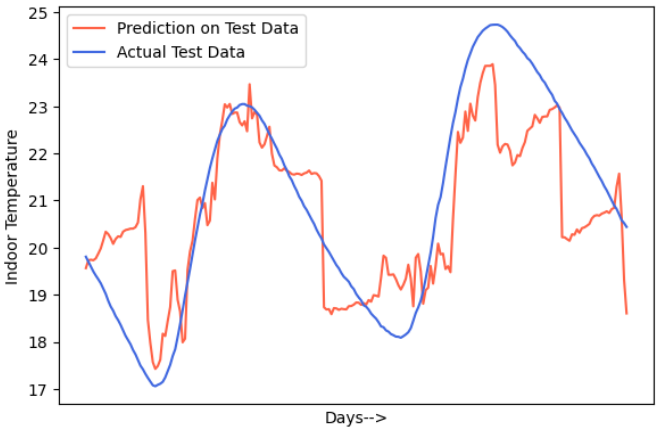

Fig. 6: Visualizing temperature forecasting results of LR model (trained on weather and date/time features) on the **test set**.

cannot have rapid changes owing to physics constraints, the lag aspects should stabilize the model for making predictions that do not have abrupt changes.

To estimate the number of lag features required we can use the partial autocorrelation values of indoor temperature with respect to its lags (previous values). Fig. 7 shows the plot of PACF values of indoor temperature with lags. According to the plot, four lag features appear to capture all of the information required for training a Linear Regression model. Model performance is very good on both training and test sets. Mean Squared Error (MSE) for prediction on training data is 0.00194 while the MSE for prediction on test data is 0.00076. Fig.8 and Fig.9 show the models prediction on test and training data.

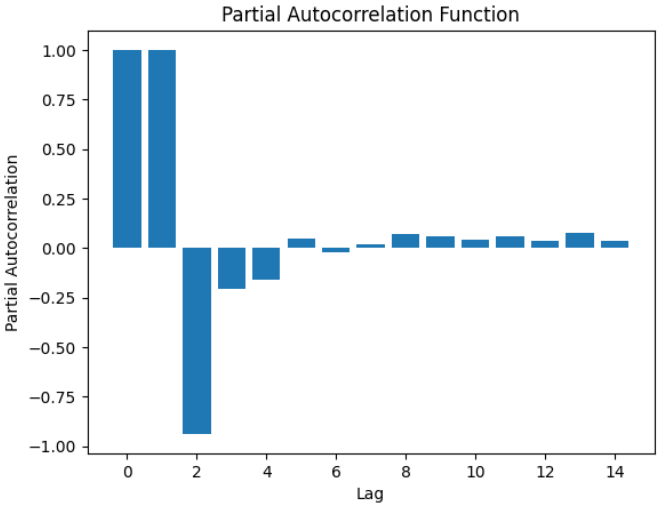

Fig. 7: Partial autocorrelation of Indoor temperature values with 15 lag features that we created.

From the plots it is apparent that LR model with lag features is very good at prediction of temperature as it have access to the previous temperature. However, the caveat here

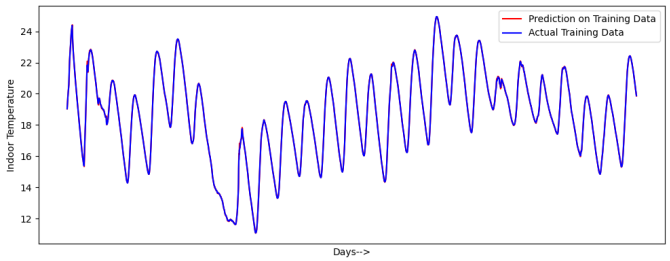

Fig. 8: Indoor temperature forecasting results (on Training data) with linear regression model trained on lag features

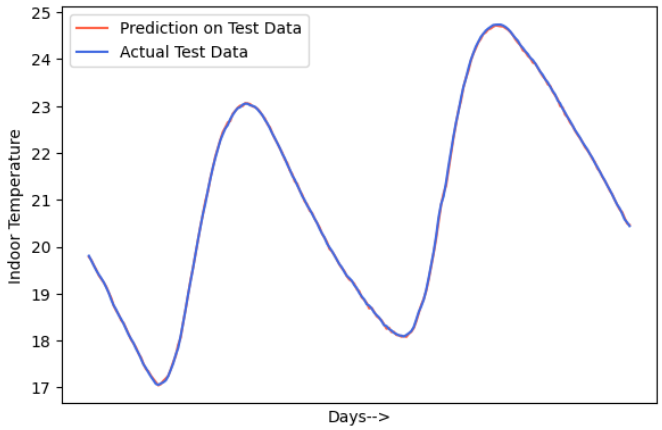

Fig. 9: Indoor temperature forecasting results (on Test data) with linear regression model trained on lag features

is that the we will not be able to predict the temperature into future for the period which we don't have the actual value of temperature to create lag features. This can be circumvented using regressive method to predict next value at every step and then considering the predicted value as actual for prediction of next value. In the aforementioned plan, however, the error in this keep piling up and our model is not able to predict very well after some time.

C. eXtreme Gradient Boosting (XGBoost)

XGBoost is an implementation of gradient-boosting decision trees, which is a very popular technique in machine learning. It is hailed as one of the best machine-learning algorithms for structured data sets. We tried using XGBoost Regressor for our problem. We attempted to fit the model to our data set using the same features as in procedure A. The mean square error on training data is 0.00028, while the error on the test set is 4.44191. The performance of the model on the training set is shown in Fig.10 and the performance on test set can be seen in Fig.11.

The charts show that the model performs well on the training data but struggles on predictions outside the data set, which is a known weakness of Tree-based estimators. This was also covered in the introduction of these modules. The MSE quantitatively indicates this constraint; we find that the error

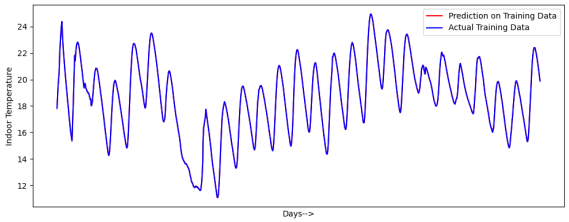

Fig. 10: Actual vs Predicted indoor temperatures obtained with XGBoost regressor on training data

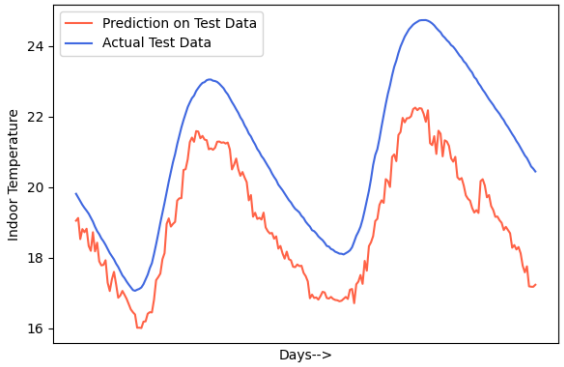

Fig. 11: Actual vs Predicted indoor temperatures obtained with XGBoost regressor on test data

on training data is relatively low, but it is significantly larger on the test set.

D. Hybrid Linear-Tree Regressor

We saw in the previous approach that the XGBoost method performed poorly outside of the training data, i.e., in extrapolation, but the model performed far better than the Linear Regression model in method A on the training data. The XGBoost plot also revealed that the model's predictions for the test set were smoother compared to method A. Therefore we try to explore a method that combines the effectiveness of non-parametric tree-based estimators with the extrapolative capabilities of linear regression. This strategy is motivated by Marco Cerliani's linear-tree package [5], which provides a solution that leverages the advantages of both methodologies.

For applying the linear tree estimator to the Smart Home's Temperature competition data, we first go through same data preprocessing steps as for Linear Regression Model. LinearBoostRegressor from the python package 'linear-tree' has been used to build our prediction model here. This model achieved a Mean Squared Error of 0.04215 on the training data and 3.26661 on the test data. This result is a improvement from the earlier mse of 4.44191 on the test data. Fig.12 and Fig.13 show the models prediction on test and training data.

The charts show that the model is very efficient at making predictions on the training data. For predictions on test data, it can generalize the dominant trend but not generate very good predictions. Our results for the Linear Tree model fall somewhere between the Linear Regression model, which had

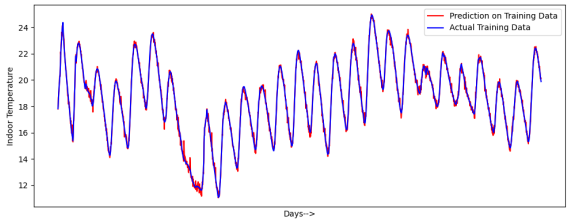

Fig. 12: Comparison of the distribution of Indoor temperatures in the training data and that of our predicted temperatures

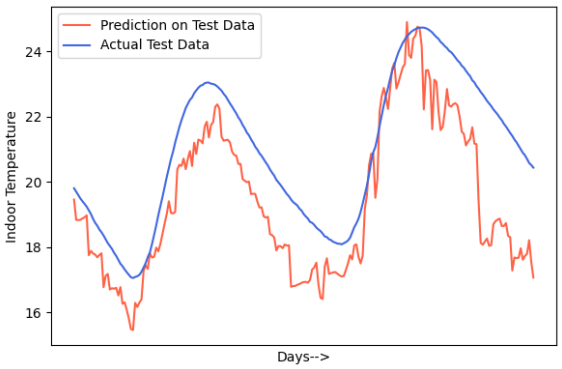

Fig. 13: Comparison of the distribution of Indoor temperatures in the training data and that of our predicted temperatures

abrupt forecasts but could generalize on extrapolation, and the XGBoost approach, which could create smoother predictions but could not capture the linear trend in the data.

IV. Conclusions

In this study, we demonstrate how machine learning methods were applied to a structured dataset in order to forecast the interior temperature of a solar-powered home. This work makes it apparent that there is no universal algorithm that can solve every problem encountered in the real world. Every algorithm has its advantages and disadvantages, and it falls upon the machine learning engineer to make strategic decisions on which algorithm to employ based on his specific application and available data. Having said that, the decision is not completely arbitrary, with an understanding of the underlying mathematical concepts behind the algorithm one can make fairly informed decisions.

In our work, we investigated the use of four different algorithms and presented the results obtained by them. Overall, the Linear Regression approach is effective at understanding long-term trends in data and is thus useful in extrapolation problems. Linear Regression armed with lag features is very efficient at handling sequential data but suffers from error accumulation when making temperature forecasts outside the known data set. While XGBoost is excellent at interpolation, it is not well suited for issues that require us to extrapolate the data, such as our time series problem. The Linear Tree method presents a middle ground between these, although it still requires tuning for each individual case.

References

[1] F. Zamora-Martínez, P. Romeu, P. Botella-Rocamora, and J. Pardo, "Online learning of indoor temperature forecasting models towards energy efficiency," *Energy and Buildings*, vol. 83, pp. 162–172, 2014, sCIENCE BEHIND AND BEYOND THE SOLAR DECATHLON EUROPE 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378778814003569

[2] S. Frandina, M. Gori, M. Lippi, M. Maggini, and S. Melacci, "Variational foundations of online backpropagation," in *Artificial Neural Networks and Machine Learning – ICANN 2013*, V. Mladenov, P. Koprinkova-Hristova, G. Palm, A. E. P. Villa, B. Appollini, and N. Kasabov, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 82–89.

[3] Kaggle, "Smart homes temperature time series forecasting," https://www.kaggle.com/competitions/smart-homes-temperature-time-series-forecasting/data, Year Accessed.

[4] C. M. Ellis, "Multivariable time series forecasting: Linear + tree," https://www.kaggle.com/code/carlmcbrideellis/multivariable-time-series-forecasting-linear-tree, Year Accessed.

[5] M. Cerliani, "A python library to build model trees with linear models at the leaves," 2022. [Online]. Available: https://github.com/cerlymarco/linear-tree#readme