Comparing the Effectiveness of Machine Learning and Statistical Methods for Forecasting Electricity Demand in NSW

Alex Fang (z5388415) Jasmine Tan (z3426586) Ruth Furness (z533449) Waikei Lau (z5349878)

Table of Contents

1. Introduction

The electricity industry has witnessed significant changes in recent years, particularly with the deregulation of the retail electricity market in Australia. As a result, New South Wales (NSW) has joined Victoria, South Australia, Queensland, and Tasmania in becoming part of this competitive landscape. Consequently, electricity retailers now compete for customers, while the Australian Energy Market Operator (AEMO) oversees the Electricity Market, a real-time

spot market or pool that matches power supply and demand through a centrally coordinated dispatch process. In this market, electricity generators sell their power to the pool, while retailers purchase electricity from the pool to sell to consumers. The dispatch price for each five-minute period is set by AEMO as the highest offer accepted from a generator.

Demand forecasting plays an essential role in power system management. Accurate forecasting helps minimize costly mistakes, such as overestimation or underestimation of demand, which can result in financial losses and system failures. The AEMO constantly balances the supply and demand of electricity since large-scale storage is not possible. To maintain equilibrium within a certain range, real-time switching in and out of supply manages fluctuations in demand. Deviations from the tolerance level caused by a significant discrepancy between demand and supply can cause considerable harm to the system. Therefore, it is crucial for market regulators and participants to have access to reliable demand forecasts.

The Australian Energy Market Operator (AEMO) regularly publishes Short Term Projected Assessment of System Adequacy (STPASA) which includes demand forecast for the next 7 days in half-hourly resolution [1], This time frame gives a balance between accuracy and practicality:

- Accuracy: Compared with hourly or daily forecast, the 7-day forecast is not too short that cannot handle sudden changes in the weather, or too long that lose accuracy and certainty.
- Planning: The 7-day forecast provide electricity generators, transmission operators, and other stakeholders with enough lead time to plan and adjust.
- Resources: It requires more resources and computational power for demand forecast more than 7 days. The balance of accuracy and cost needs to be considered for practicality.

Electricity demand is influenced by various factors, such as temperature, humidity, wind speed, GDP, and population [2]. Temperature is the most common variable in forecasting electricity demand [3].

Our analysis is based on a comprehensive dataset comprising three years of electricity demand and temperature observations. The data was pre-processed and cleaned to ensure its accuracy and reliability for further analysis.

In this report, we utilized both statistical models like ARIMA, and machine learning techniques, including neural networks and random forest, to develop demand forecasting models that capture the unique features and trends of the NSW electricity demand in recent years. We have done a comprehensive comparison based on the result from different models, analysed the strength and weakness of each model. We have also done some experiment of combining statistical model and machine learning techniques to generate a hybrid model for a more reliable and accurate performance.

Overall, this research highlights the potential of using advanced data analytics and machine learning techniques for enhancing the resilience and efficiency of the energy sector.

2. Literature Review

The relationship between electricity demand and temperature is a critical aspect of energy demand forecasting. The existing literature has addressed this relationship using various methodologies, recognizing the non-linear nature of electricity demand as a function of temperature [4].

A common approach involves using heating degree days (HDD) and cooling degree days (CDD) variables to capture the non-linearity [5]. However, to consider the temperature directly in the model may make more sense for the analysis of sensitivity of electricity demand to temperature.

An alternative approach to estimate the non-linear relationship between electricity sales and temperature is to use non-parametric or semi-parametric models [6]. The disadvantage is that it does not specify the relationship between electrical demand and temperature.

Several studies have developed forecasting models of electricity demand at various resolutions, focusing on daily, monthly, and hourly data [7][8][9]. For instance, De Felice et al. (2013) demonstrated the significant improvement of prediction performance when incorporating operational weather forecasts for daily electrical load during summer working days.

On the other hand, studies such as Bessec and Fouquau (2008) [10] and Lee and Chiu (2011) [4] investigated the relationship between temperature and electricity demand, using smooth transition models with monthly and yearly data.

Traditional statistical and regression methods have been widely adopted in the field of energy demand forecasting. Hyndman, R. J., & Athanasopoulos, G. [11] applied an extended ARIMA model to do the time series forecasting for the electricity demand by temperatures. It used a regression model with ARIMA errors and allowed the inclusion of external variables in the model. This approach combines the advantages of time series and regression models by allowing the inclusion of external variables while also capturing time series dynamics. To make forecasts using a regression model with ARIMA errors, both the regression and ARIMA components of the model need to be forecasted separately and then combined. To obtain these forecasts, the predictors must also be forecasted. This fits our purposes to consider both time periods and temperature variables when forecasting the electricity demands.

Wang and Bielicki [12] applied a segmented ARIMA technique to study the effects of different weather conditions on electricity demand. They developed three segmented ARIMA models: a temperature-only model, a model with additional weather variables, and a two-step model to examine the effects of relative humidity. They found that cooling and heating degree hours, previous days' temperatures, and relative humidity all influenced demand, and that demand is affected by some acclimation to current weather conditions. Mean absolute percentage errors were used to determine effectiveness of the models, with the full model typically having the lowest values. Based on their findings, a segmented ARIMA technique with additional weather variables will be implemented to forecast demand.

But there has been a growing trend toward the use of machine learning methods. This has been evidenced by the increased adoption of machine learning by entities such as the Australian Energy Market Operator (AEMO) [13].

In the study conducted by Shin and Woo titled "Energy Consumption Forecasting in Korea Using Machine Learning,"[14] the effectiveness of three models, namely Random Forest (RF), XGBoost (XGB), and Long Short-Term Memory (LSTM), were evaluated in predicting energy consumption. The findings of the study revealed that both LSTM and RF models performed

well in different time periods. Specifically, LSTM was found to be more effective before the COVID-19 pandemic when energy consumption was on an upward trend, while RF was found to be more effective after the pandemic when energy consumption was decreasing.

The results of this study illustrate the real-world applications of RF, XGB, and LSTM methods in energy demand forecasting, which makes them highly relevant to the current project. Moreover, the authors of the study also observed that while econometric approaches may be more effective in predicting energy consumption when there is less irregularity in the time series, machine learning techniques, such as those employed in their study, are better suited for handling unexpected and irregular time series data. These observations highlight the potential benefits of using machine learning methods in energy demand forecasting, particularly in scenarios where the time series data is unpredictable or irregular.

Overall, the existing literature has explored various methodologies to figure out the relationship between electricity demand and temperature. While significant progress has been made, there is still need for improvement to build more accurate models and to make explicit evaluations of the impact of meteorological variables on electricity demand.

3. Material and Methods

i. Software

The team has utilized different software for different tasks in this project trying to maximize the strength of each. For the modelling part, we mainly use R for the statistical model building, including ARIMA and GAM, and Python for the machine learning models, including random forecast, LSTM and XGBoost. For the visualization, we also used one of the leading business intelligence tools – Microsoft Power BI for some of the data exploration as well as consolidating and visualizing the forecast from different models. We also used github to store and share the code and relevant data.

ii. Data Description

Two sets of data have been used in this project, including the total electricity demand of New South Wales measured in five-minute increments sourced from the Market Management System database, the air temperature measured in thirty-minute intervals at the Bankstown Airport weather station sourced from the Australian Data Archive for Meteorology.

While the presence of common data variables such as date, time, and location across the three datasets enables the matching of temperature observations with demand actuals and forecasts, it is worth noting that the temperature and demand data were recorded at different frequencies, with temperature data measured in 30-minute increments and demand data measured in 5-minute increments. Given the different frequencies at which the temperature and demand data were recorded, it is essential to address this issue during the pre-processing phase to ensure that the data is properly aligned and can be accurately analyzed.

iii. Pre-Processing

Data outliers

During the pre-processing of the data, it was observed that some temperature readings in the year 2011 showed significant deviation from the normal trend as shown in Fig. 1. Subsequent investigation identified these readings as data errors that were not representative of the usual trend. Therefore, to ensure the accuracy of the data, these readings were excluded during the pre-processing phase. This step ensures that the subsequent analyses and modelling are based on accurate and reliable data.

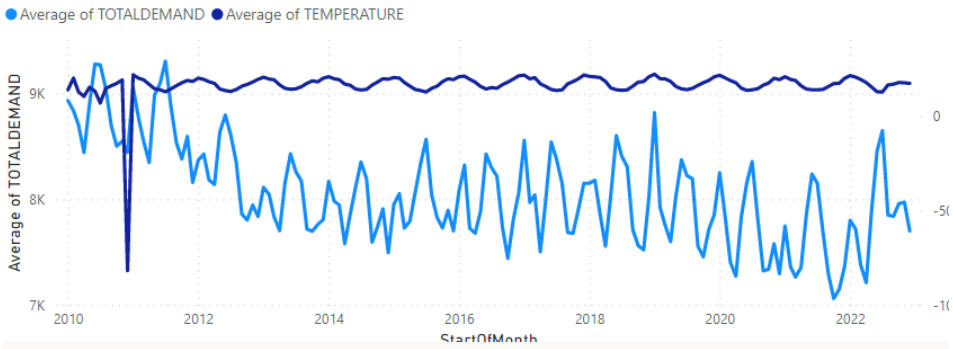


Fig. 1. Inspection of the Demand and Temperature data.

Data duplicates

In addition, during data pre-processing, it was discovered that some timestamps in the demand data were duplicated. To ensure the accuracy of the analysis, the duplicated demand was removed from the dataset. This step ensures that the data used for further analysis and modelling is free from any inconsistencies that could potentially affect the results.

Data join

To account for the differences in the frequencies of the demand and temperature data, the datasets were joined using a five-minute frequency. Missing temperature values were filled in with the most recent recording to ensure the data is properly aligned. Both demand and temperature are aggregated to 30-mins by averaging the data from the first 5 minutes to half hour, and 35 minutes to the start of next hour. This approach ensures that the data is properly formatted and suitable for further analysis and modelling.

Time frame of analysis

Our research team arrived at a consensus to employ the most recent three-year dataset, starting from 2019, as it is deemed most reflective of the prevailing economic and climate conditions during the COVID-19 pandemic. The intention behind this approach was to develop models that capture current trends of the electricity demand in New South Wales. This selection criterion is underpinned by the understanding that recent data exerts a stronger influence on modelling and prediction accuracy, thus elevating the quality of our research findings.

Training and test set

For regular dataset, we can randomly reserve part of the data as the test set, however, time series is different as the data points need to be consecutive. We aim to utilize a training dataset spanning two years, from August 1st, 2019, to July 31st, 2021, to develop our model.

Subsequently, we plan to assess the efficacy of the model using a distinct dataset that covers a one-year period from August 1st, 2021, to July 31st, 2022. Our primary objective is to generate a seven-day demand forecast. However, we recognize that the demand patterns may vary across different weeks and seasons. Hence, by allocating one year as the test set, we aim to gain a more comprehensive understanding of the model's performance across a more extended period, encompassing various seasonal conditions. Our goal is to forecast nearly one year's demand using each model and conduct a comprehensive evaluation of their performance.

iv. Feature Engineering

The secondary aim of this study was to optimize the value of the research for market participants by minimizing the number of data sources and requirements, while simultaneously maximizing forecast accuracy based on readily available data. This approach had three major benefits.

- Firstly, it significantly reduced the complexity of the models, resulting in more parsimonious models that are more likely to be adopted by market participants.
- Secondly, it increased the interpretability of the results, making it easier for decision makers and analysts to understand the findings and implications of the research. By making the data sources and analysis methods accessible and easy to understand, this study has the potential to drive innovation and inform decision-making in the energy sector.
- And thirdly, it provides a basis from which to add additional variables as relevant to the changing market conditions, making it a customisable benchmark that can be adapted to different contexts and use cases – some of which are considered below.

The following features were incorporated into the subsequent models, and their associated rationale for their inclusion is presented below:

- Seasons - The relationship between energy demand and temperature is well established, particularly during periods of warmth and cold that characterize the winter and summer months.
- Holidays - We posit that electricity demand during holiday periods may significantly differ from typical weekly usage patterns.
- Month - The month variable provides greater temporal granularity than seasons and can be used interchangeably with the seasons feature.
- Day of Week - A weekly pattern that distinguishes energy consumption during weekdays from that during weekends.
- Hour and Half Hour - A daily pattern that distinguishes energy usage throughout the day, influenced by rising temperatures during the day and offset by peak photovoltaic energy generation during midday.
- Lagged variables of temperature and energy demand were added to capture the impact of past values on future energy demand. By including these variables, the model can consider past trends and seasonality, leading to improved forecasting accuracy. This technique is useful for predicting future energy demand.

The following are optional variables that may be considered to customize energy demand forecasting and meet user-specific requirements:

- Firstly, the Hourly Solar variable serves as an indicator of Photovoltaic capacity during periods of high sunshine, which offsets power demand during midday.
- Secondly, Extreme Weather Events can act as a leading indicator for decreased power supply usage, owing to reduced industrial activities.
- Lastly, the inclusion of Smart Homes, Electric Vehicles (EVs), and Virtual Power Plants as variables highlights structural changes in long-term power demand.

It is crucial to consider these optional variables while forecasting energy demand, as they enable more accurate predictions and cater to user-specific requirements.

v. Measure of accuracy

We use Mean Absolute Percentage Error (MAPE) to measure the accuracy of the forecast from all models. MAPE is a commonly used metric as it shows the error percentage and very easy to understand. Prediction interval is important in measuring the forecast, however it is challenging to produce that in some machine learning models like LSTM, so it will not be our key measure when we compare the model accuracy. But we will show the prediction interval in the ARIMA and Random Forest section.

4. Exploratory Data analysis

First, we take a quick look at the demand and temperature data provided in Fig. 2. We can see that temperature has been very consistent and stable over the 12-year period, while the demand shows a quick drop in 2012, which was mainly driven by the closure of Kurri Kurri aluminium smelter. The demand has been stable at similar level until late 2019, it seems the demand decreased again, and the pattern changed a bit, possibly driven by covid.

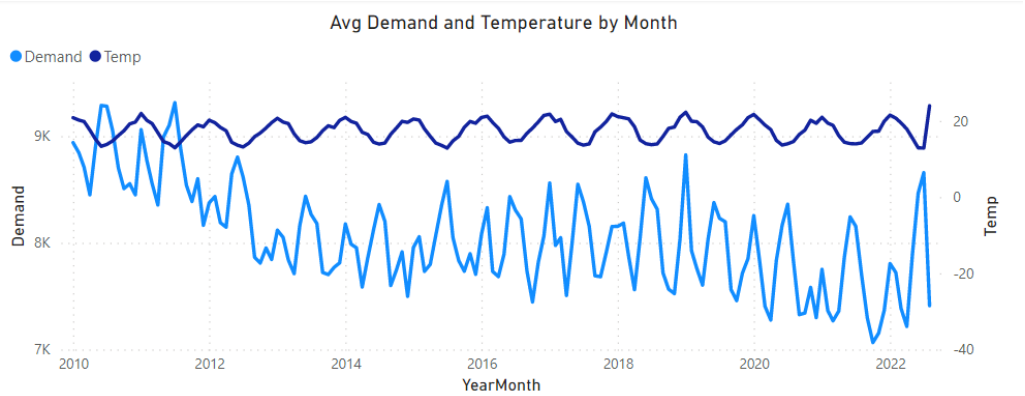


Fig. 2. Trend of the demand and temperature after some cleaning.

Next, we will look at the correlation of the demand and demand in Fig. 3. For each calendar year, we plot average demand and temperature by time of the day as below. We can see overall they shape the same in a day, however a very interesting thing is that they become negatively correlated in the mid- day, and this negative correlation became more and more obvious over years, which is highly likely driven by the increasing penetration of the rooftop solar panels, which effectively reduce the operational demand that customers require from the grid. This

also suggest that the data from recent years may be more relevant in terms of building models for demand forecasting today.

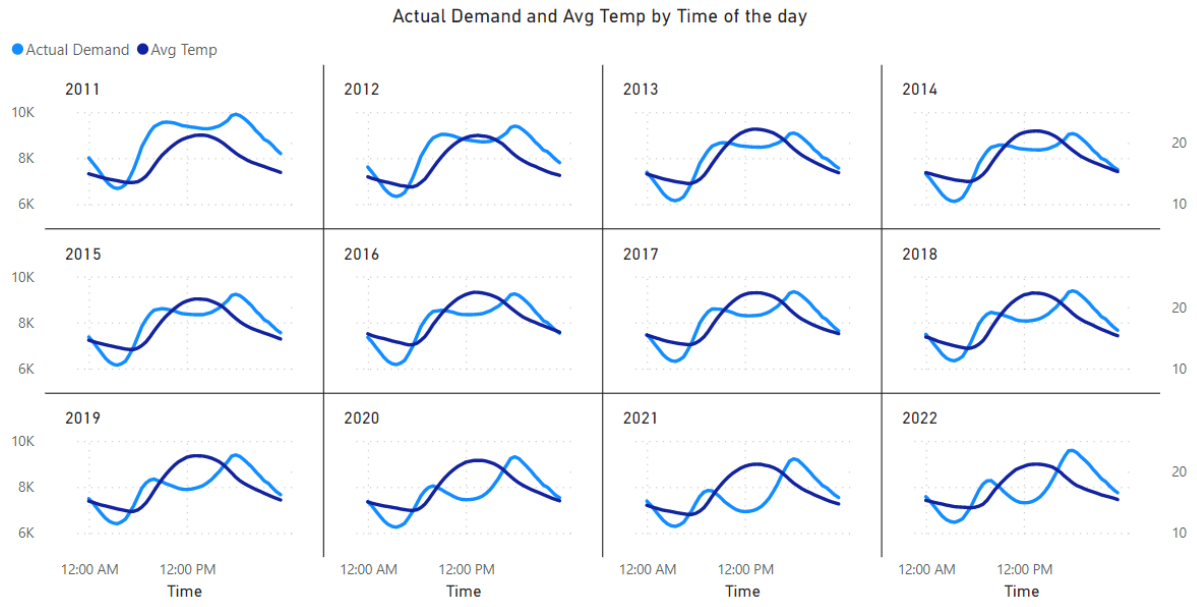


Fig. 3. Demand and Temperature correlation by time of a day over years.

We know that demand can be quite different month by month, and box plot is good way to show the range and variations between months. As we can see from Fig. 4, the winter months like June and July have the highest demand, followed by the summer months, and the shoulder seasons (spring and autumn) have the least demand in a year. Another interesting thing is that the hot months from November to February, the outliers are significantly more than other months, the demand can spike to 14,000 MW, which is the highest in a year. On the other side, winter demand seems very stable with very few outliers. The implication of the observation is that it might be more challenging to forecast summer demand, and it's more sensitive to weather changes, extreme hot weathers in specific.

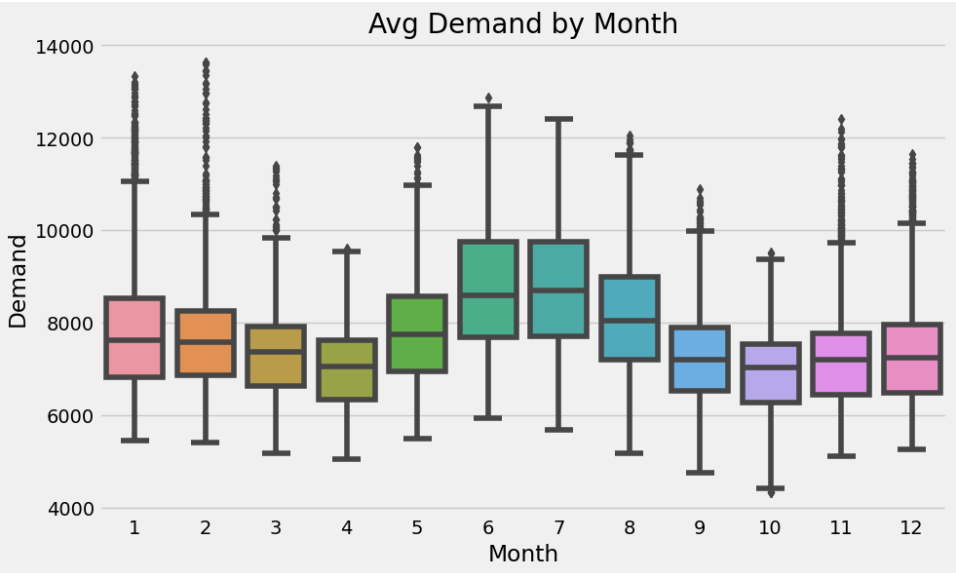


Fig. 4. Demand distribution statistics by month.

Now let us zoom in to a week from late July 2022 as shown in Fig. 5. It gives us some clue about how the demand changes over a week. It started weak but started to ramp up on Monday, peaked on Tuesday and kept at high level but gradually reclining for the next three weekdays, and the demand dropped quite a lot over weekend.

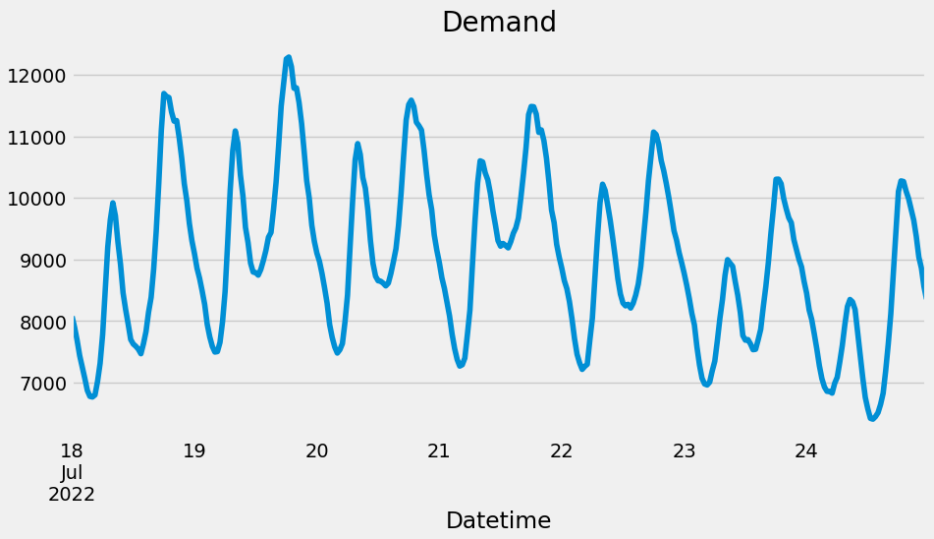


Fig. 5. A week demand trend/shape from late July 2022.

Now we further zoom into a day, but layer in the different seasons in Fig. 6. It gives more insights into the demand shape in different period. First the spring and autumn are in similar shape, and the volumes are very close in the mid night and evening, the differences are just the mid of the day Autumn has higher demand as the weather gets colder. They have a typical "duck curve", where there are two peaks in a day. Morning peak is around 7 AM when people get out of the bed, turn on the light and toasters. A bigger peak is in the early evening around 6PM when most people get home and prepare dinner. Winter follows the same shape with more distinct peaks and demand is much higher throughout the day with the heating consumption. Summer shows the same curve in the evening with the shoulder seasons but the pattern in the day is unique, which didn't decrease, which sort of making sense. When it gets warmer in the other seasons, people may turn off the heating which reduce the consumption, but in summer people would turn on the air conditioning which increased the demand.

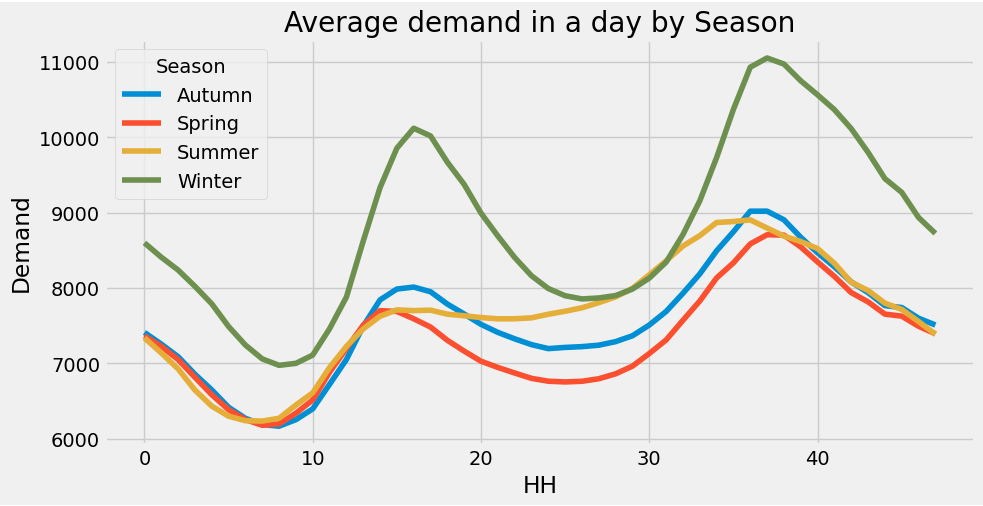


Fig. 6. A typical demand curve in a day by season.

Previously in the week trend we have noticed that weekend demand is lower than workdays, now let us plot them together to see how different they are, meanwhile, we split holiday into a separate bucket. As shown in Fig. 7, demand in workdays is significantly higher, and holidays are further less than weekend, especially in the morning peak, when people do not need to get up early to meet some routine schedules.

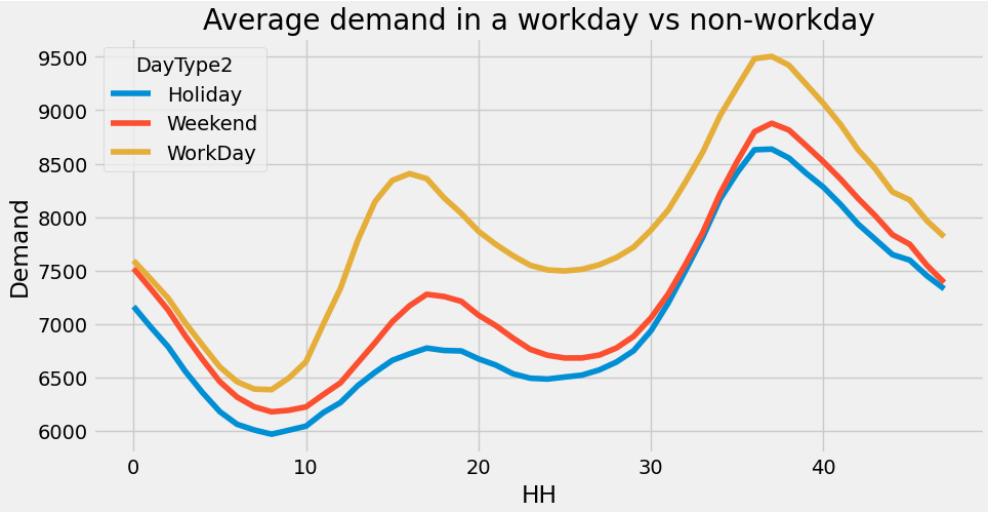


Fig. 7. A typical demand curve from different day type.

5. Modelling and Result

i. Linear Regression:

Based on the explorational analysis, on top of the temperature, we added some time features into the dataset as explanatory variables and try to build a simple linear regression model. It provides us some baseline to compare with other more sophisticated models we build. The intercept and coefficients of this linear model are shown in Fig. 8. As we can see, seasons and day type (holiday, weekend, and workday) have much bigger influence than any other variables in this model.

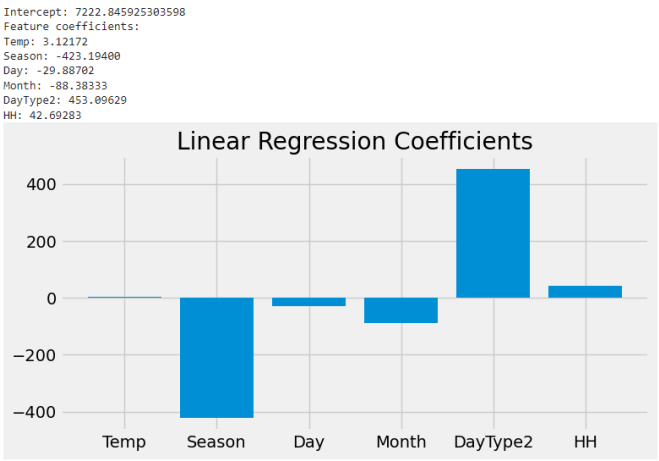


Fig. 8. Coefficients from linear regression model.

The overall Mean Absolute Percentage Error (MAPE) is about 11%, and the forecast for the first week of the test set is shown in Fig. 9. Apparently, the linear model captured some basic patterns, like weekend consumption is lower, and the time of the peak demand etc. However, as it can only interpreter the relationship linearly, it is far away from being a robust model and is not capable of providing accurate forecast at a high granularity.

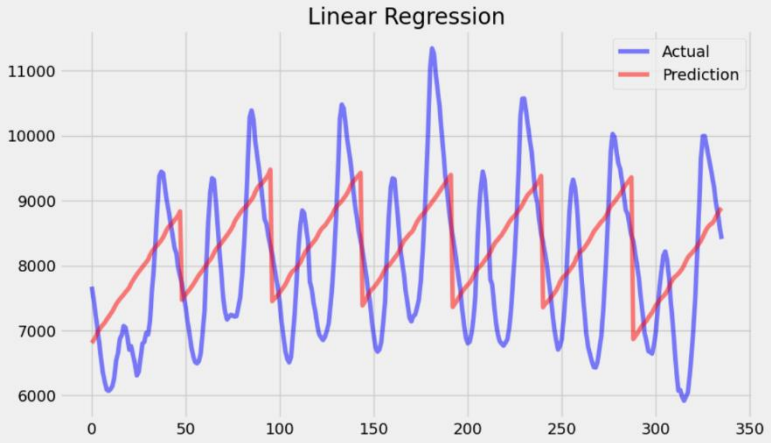


Fig. 9. Linear regression forecast compared with actual demand.

ii. GAM:

We can see that the relationship between the time features, temperature and demand is complex and hard to be captured through linear regression. we tried another statistical model - The Generalized Additive Model (GAM). GAM is still a regression-based model, but it is very flexible by producing many splines and put them together [14], as a result, it can better capture the non-linear relationship in the dataset.

As our data is a time series, it would be also helpful to add some lagged variables. These lags are basically what happened in the past, and they can be highly correlated to what could happen next as demand follows some basic seasonal, weekly, and daily pattern. We added four lags here with the demand from one week before, two weeks before and so on.

Then we fit the GAM model with the train set and predict the first week in the test set.

As the forecast shown in Fig. 10, it generates more smoothed curves that fit our data better, and the MAPE is about 8%, which is almost 30% better than linear regression.

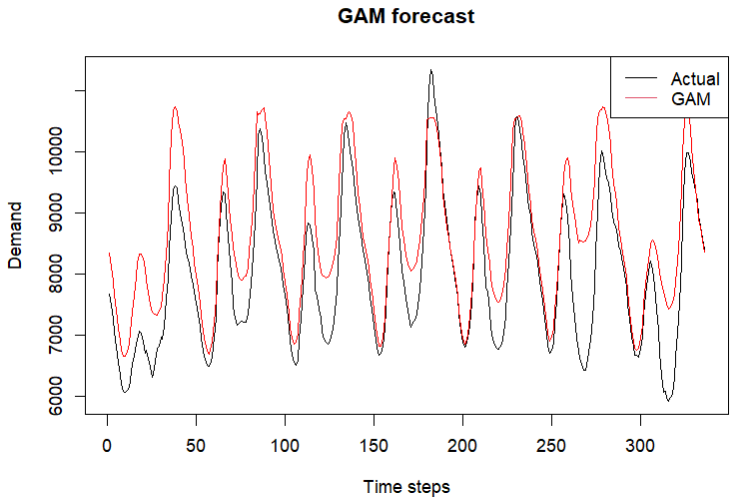


Fig. 10. Forecast from generalized additive model for the first week of August 2021.

iii. ARIMA

Autoregressive models (AR) are self-regressing models that represent the output as a linear combination of past values in a time series. The observed measurements are used for the estimation of autoregressive coefficients. AR models can be joined with moving average models (MA) to make an autoregressive integrated moving average (ARIMA) model (16).

The main benefit of the time series method is its simplicity and ease of use, based on the assumption that past patterns of a variable will impact the future. But this approach has disadvantages since it ignores possible variable interactions.

In ARIMA models, the autoregressive (AR) part means that the changing variable is regressed on its own previous values. The moving average (MA) part shows that the regression error is a linear combination of prior error terms. The integrated (I) part means that the data is changed with the difference between their values and the preceding values.

When combining an autoregressive process based on prior values, i.e., order p, with a moving average process of prior errors, i.e., order q, a general ARIMA model is formed with order (p, q). To work with the nonstationary data over time, a difference filter, i.e., order d (the degree of differencing), is used to transform the data so it becomes steady, to develop forecasting models. The common notation is ARIMA (p, d, q).

As ARIMA modelling is commonly used for time series modelling and energy demand forecasting, it was chosen not only as a good baseline model for comparison to the machine learning methods also being used, but as what could be a good model in its own right.

ARIMA modelling assumes that time series data is stationary, this means that the statistical properties of the data do not depend on the time of observation and that there are no trends or seasonal patterns[1]. Because electricity demand displays seasonal trends, such as peaks at certain times of the day or lower demand on weekends, these must be removed before

[1] https://otexts.com/fpp3/stationarity.html

modelling. This is accomplished through differencing and is the integrated part of the ARIMA model.

There are a number of tests to determine the amount of differencing required to make time series data stationary. However, the R functions, `auto.arima` (forecast package) and `ARIMA` (fable package) are both capable of running these tests and determining this automatically, and so these were used throughout the modelling process to save time and ensure consistent model selection.

An R time-series object was used to fit a model predicting demand (with demand as the sole predictor) using the auto.arima function, with frequency 48 x 7, to capture the variation in demand throughout the 7-day week. The function is able to include the serial dependence structure of the errors and get the independent residuals. It also minimizes the AICc to find the best error structure. The final model chosen by the `auto.arima` function was an ARIMA (3, 0, 2)(0, 1, 0)[336] model. The model has no non-seasonal differencing and was seasonally differenced once.

Model coefficients and standard errors can be found in Table I below.

TABLE I. MODEL COEFFICIENTS AND STANDARD ERRORS FOR ARIMA (3, 0, 2)(0, 1, 0) MODEL

	ar1	ar2	ar3	ma1	ma2
Coefficients	0.8842	0.7781	-0.6879	0.5763	-0.2809
Standard Errors	0.0646	0.1121	0.0488	0.0641	0.0171

Visual inspection of the actual and fitted values and a one week forecast indicate that the model is capable of capturing typical changes in demand throughout the day, e.g., peaks during high demand time periods (Fig. 11).

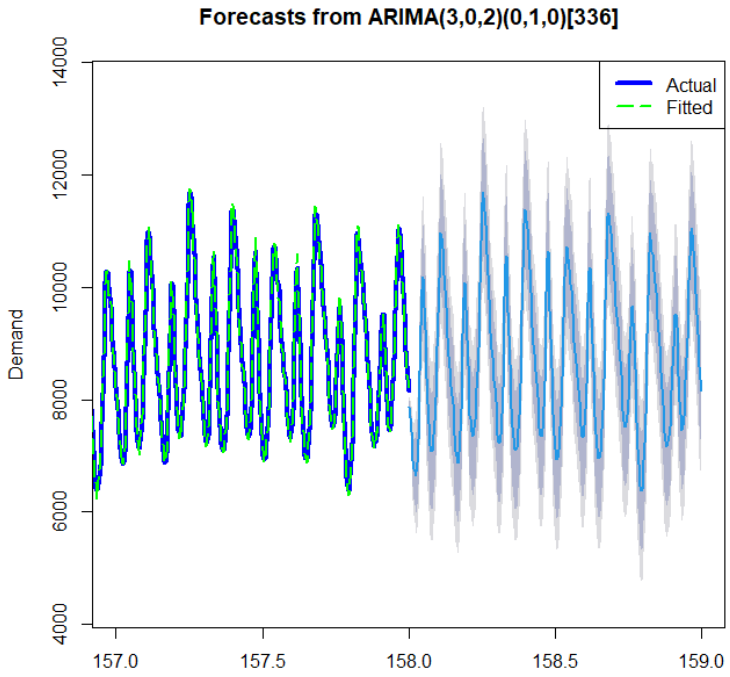


Fig. 11. Plot of actual and fitted data from final week of training set with one week of forecasted data from ARIMA (3, 0, 2)(0, 1, 0) model. 80% and 95% prediction intervals are shaded.

While the `auto.arima` function appears to have produced a good model, it was time-consuming to fit, and this was expected to worsen with an increase in model complexity, i.e., the addition of external variables. The use of alternative functions was explored, including the `arima` function (stats package) and `Arima` (forecast package), however models produced were often incomplete, with either coefficients or standard errors not estimated, and thus these were deemed unsuitable.

As previously mentioned, the `ARIMA` function in the fable package, like the `auto.arima` function, is able to determine the best model dimensions automatically and model fitting with this function was much more reliable than with other functions. Thus, it was used to fit all future models.

Once again, an ARIMA model was fit predicting demand with demand as the only predictor. Fourier terms were included to capture the seasonality within the data daily, weekly, and yearly.

- Daily seasonality was included because of the pattern of demand across the day, in which there is, on average, a peak in demand in the mornings and another peak in demand in the evenings.
- Weekly seasonality was included because of the pattern of demand throughout the week, in which, on average, days during the working week see higher demand than days during the weekend.
- Yearly seasonality was included because of the pattern of demand across the different seasons throughout the year, with the winter and summer months displaying, on average, higher demand than autumn and spring months.
- A monthly seasonality term was also considered, however, there is no clear pattern within the month to justify such an inclusion.

The orders of each of the Fourier terms were first selected based on those given within Chapter 12.1 of *Forecasting: Principles and Practice* (3rd ed)[2].

Following this initial model, five other ARIMA models were fit, each including different combinations of a selection of external variables: half hourly average temperature, whether it was a public holiday, and what season of the year it was (model specifications can be found in Table II).

Dummy variables were used for the inclusion of public holidays and seasons. For public holidays, 0 indicated it was a normal day and 1 indicated a public holiday. For the seasons, three dummy variables were used, one for each of Autumn, Winter, and Spring, with the variable being encoded as 1 if the observed demand data point was within the respective season, and all variables being encoded as 0 to indicate Summer.

Exploration of the relationship between electricity demand and temperature, as shown in Fig. 12, indicated the need for a non-linear model. Since demand increases at both high and low temperatures, this suggests the need for the inclusion of a quadratic term to capture the

[2] Code excerpt: `fourier(period = "day", K = 10) + fourier(period = "week", K = 5) + fourier(period = "year", K = 3)`
https://otexts.com/fpp3/complexseasonality.html#example-electricity-demand

relationship appropriately. Transformation of the data was also considered; however, a suitable transformation could not be determined. Thus, temperature was included in the model based on code given in Chapter 10.3 of *Forecasting: Principles and Practice* (3rd ed)[3].

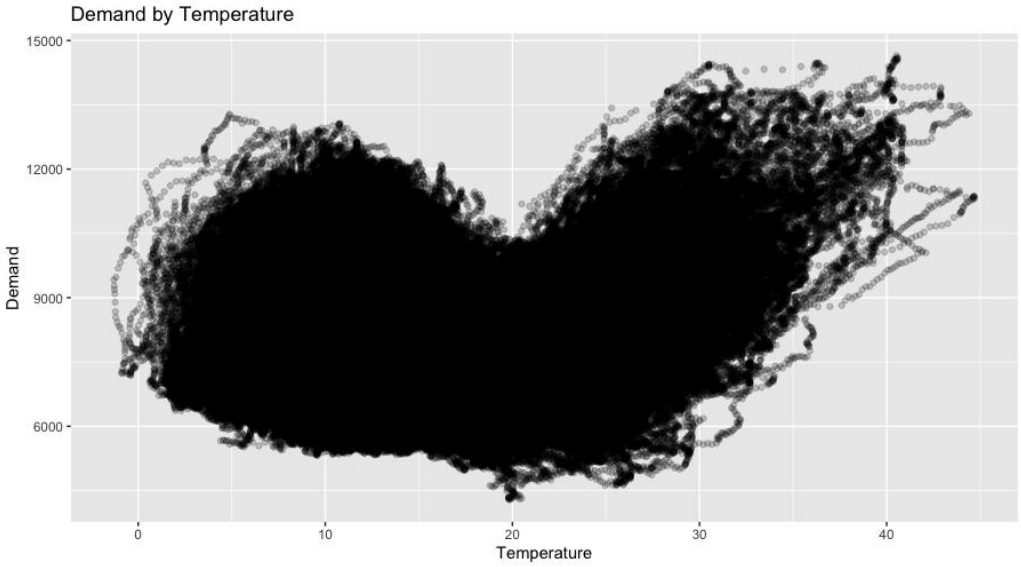


Fig. 12. Plot of relationship between Demand and Temperature from 2010 to August 2022.

Two one-week forecasts were produced for each model and their MAPE values were compared (Table II) to help determine whether the inclusion of any external variables improved the model. A visual inspection of the forecasts for one of the weeks versus the actual values was also undertaken (Fig. 13).

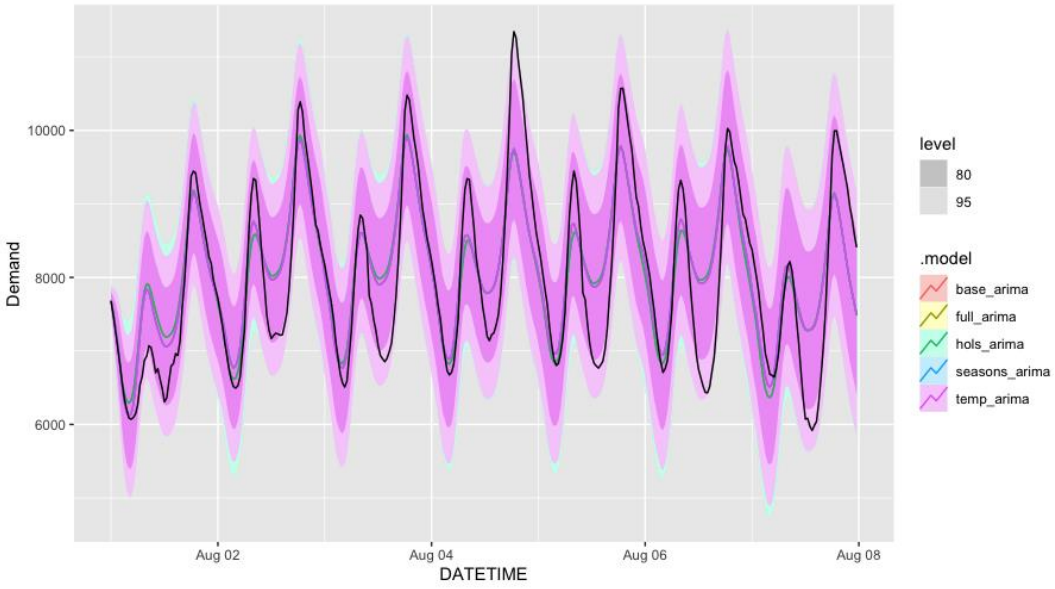


Fig. 13. Plot of one week of forecasted data from ARIMA models fit with fable package. 80% and 95% prediction intervals are shaded. Forecast for 1st-7th August 2021.

[3] Code excerpt: `ARIMA(Demand ~ Temperature + I(Temperature^2)`
https://otexts.com/fpp3/forecasting.html#example-forecasting-electricity-demand

TABLE II. MAPE 1 FORECASTS WERE FOR AUGUST 1ST-7TH 2021, MAPE 2 FORECASTS WERE FOR FEBRUARY 1ST-7TH 2022, MAPE 3 FORECASTS WERE FOR AUGUST 1ST 2021-JULY 31ST 2022.

Model	MAPE 1	MAPE 2	MAPE 3
Demand (3,1,2)(0,0,1)[2]	6.72	8.92	7.50
Demand, Temperature (3,1,2)(1,0,0)[2]	6.38	10.4	7.64
Demand, Public Holidays (3,1,2)(0,0,1)[2]	6.72	10.5	7.51
Demand, Seasons (3,1,2)(1,0,0)[2]	6.72	9.75	8.95
Demand, Temperature, Public Holidays, Seasons (3,1,2)(1,0,0)[2]	6.38	11.2	7.43
Adjusted Demand (3,1,2)(0,0,1)[2]	6.80	10.3	8.89
Adjusted Demand, Temperature (3,1,2)(0,0,1)[2]	6.47	12.7	7.55
Demand, Temperature, Public Holidays (3,1,2)(0,0,1)[2]	6.37	9.89	7.80
Demand, Segmented Temperature, Public Holidays, Seasons (3,1,2)(0,0,1)[2]	6.18	24.6	25.8

The MAPE values suggest that the inclusion of external variables had varied effects on the model. The comparison plot seems to show that the model containing temperature was better at predicting the highs and lows of energy demand.

The model with some of the highest MAPE values for the August forecasts and one-year forecasts was the one containing only the seasons as a predictor. Thus, a model containing temperature and public holidays was also fit to explore whether these two variables together could produce a better model. The orders of the Fourier terms were also adjusted[4] in an attempt to improve the model's ability to predict seasonal patterns (model specifications and MAPE values can be found in Table II).

Overall, none of the models appear clearly better than any of the others. The plot of the forecasted week (Fig. 14) for the three new models shows very little difference between them, perhaps with the ARIMA model with just demand and adjusted Fourier terms performing slightly worse than the other two.

[4] Demand model: fourier("day", K = 12) + fourier("week", K = 6) + fourier("year", K = 4)
Temperature model: fourier("day", K = 12) + fourier("week", K = 5) + fourier("year", K = 4)

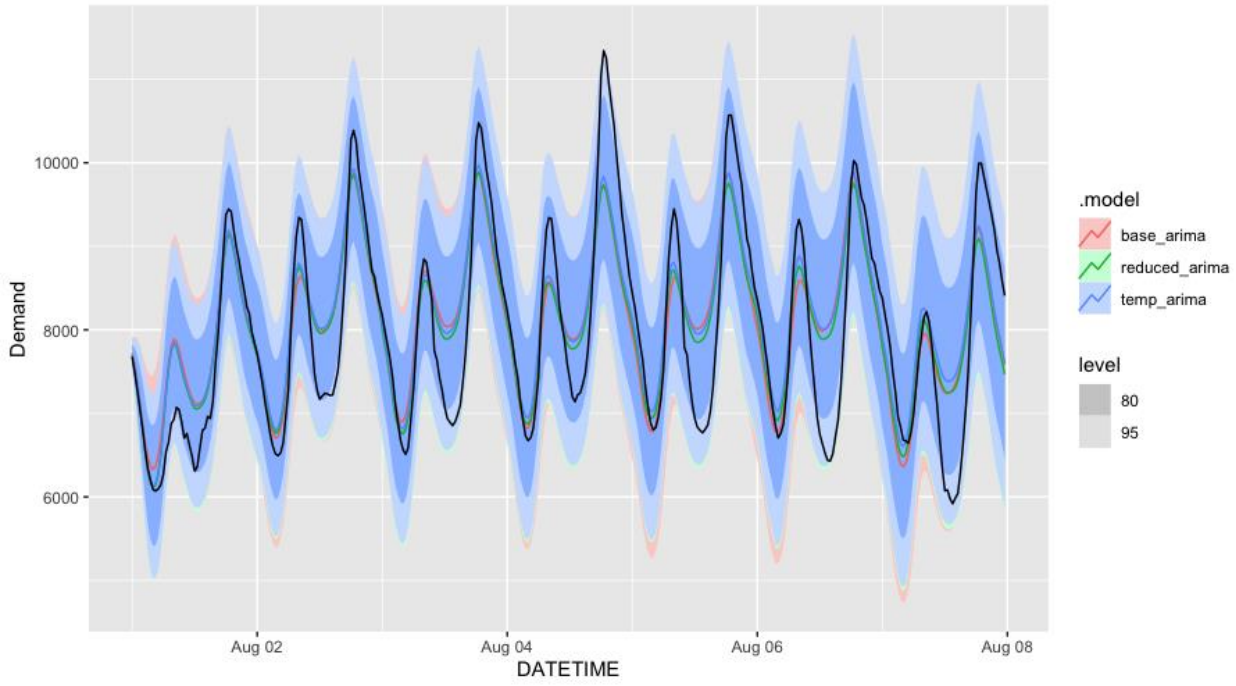


Fig. 14. Plot of one week of forecasted data from adjusted ARIMA models. 80% and 95% prediction intervals are shaded.

Given the goal of this research is to explore what kind of models can most accurately forecast one week of energy demand, emphasis could be placed upon how the models perform within the one week forecast that directly follows the training data. In this case, the models including temperature as a predictor typically had the lowest MAPE values. Of these, the model with all the external variables and the model with the adjusted Fourier terms have the lowest MAPE values for the one-year forecast.

A segmented approach to modelling was also to be undertaken, however, following model fitting with the fable package it was found that the model object produced was incompatible with the `segmented.arima` function (segmented package). Instead, the model was fit with a piecewise linear function of temperature, with knots determined by the segmented function at 7.66ºC and 21.65ºC (2dp), implemented similarly to the example in Chapter 12.1 of *Forecasting: Principles and Practice* (3rd ed)[5].

While this segmented model performed slightly better than any of the previously fit models on the one-week August forecast, it was much worse with the one-week February forecast and the one-year forecast (MAPE values in Table II). This may have been due to incorrect model specification or sub-optimal knot selection (including the use of too many or few knots). Time constraints did not allow for further investigation.

[5] Code excerpt: `mutate(Cooling = pmax(Temperature, 18))`
https://otexts.com/fpp3/complexseasonality.html#example-electricity-demand

iv. Random Forest

Why random forest?

The traditional use of random forests has been in classification tasks, for which they are highly suited. However, more recently, an increasing body of research has explored the application of random forests in time series analysis and its efficacy in multi-output and chain regression applications. [20]

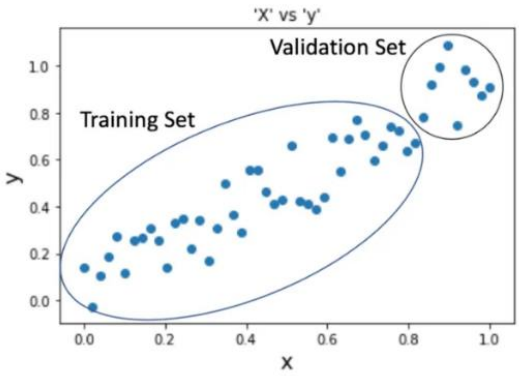
The demand for energy is characterized by variations on an hourly, daily, weekly, monthly, seasonal, and meteorological basis, as well as solar exposure and other factors. Therefore, it can be inferred that energy demand can be categorized by numerous explanatory variables, making it an excellent candidate for decision tree algorithms such as random forests. Decision tree algorithms, such as random forests, segment the data into subsets or "nodes," which enable a more nuanced evaluation of the features that contribute to a particular prediction.

Assumptions

Without imposing any formal distributional assumptions, Random Forests are considered non-parametric and therefore capable of accommodating skewed and multi-modal data, as well as categorical data that are either ordinal or non-ordinal. This property makes the algorithm versatile and suitable for various types of data, which can be especially useful in time series applications.

Key considerations

Covariate stationary



When dealing with new data, Random Forest models can only provide predictions based on the average of training values seen before. If the validation set contains data points that exceed or fall below the range of training data points, Random Forest models may generate average outcomes as they lack the ability to extrapolate and understand the growth or decline trends in the data. Consequently, Decision Trees fail to deliver accurate results for out-of-distribution data. In regression against time, every future point in time can be considered out-of-distribution. [21]

To address this issue, a more promising approach involves treating the future of a random variable as dependent on its past realizations, accomplished through the pre-processing step of adding lagged predictors. However, this alone does not resolve the extrapolation problem. To address this final challenge, the trend needs to be removed first. The model can then be fitted, the time-series forecasted, and the forecast "re-trended." One approach to removing the trend is to use first-differences, which involves transforming the time-series via first-order differencing. In addition to the deterministic trend, this method can also remove stochastic trends. [22] [23]

Multiple time horizons

A multi-output problem refers to a supervised learning task where multiple outputs are to be predicted, with Y representing a 2-dimensional array of shape (n_samples, n_outputs). In cases where the outputs are uncorrelated, a simple approach involves constructing n independent models, with each model dedicated to predicting a distinct output. However, given that the output values associated with the same input are often correlated, a more effective approach involves building a single model that can concurrently predict all n outputs. This approach has the advantage of requiring less training time, as only one estimator is constructed, and can result in higher generalization accuracy.

When exploring the utility of random forests, it is noteworthy that this algorithm natively incorporates a solution for addressing multi-output problems. Specifically, Scikit-Learn's Random Forest regressor is capable of learning the dependence structure not only between inputs and outputs but also among the outputs themselves. By implementing this approach, random forests can achieve superior performance in multi-output problems, with the added advantage of reducing the risk of overfitting. [24]

Time series to supervised learning

The provided data set is a time series, but it must be transformed to suit supervised learning. One potential method is an auto-regressive approach, which considers future values of a random variable to be contingent on its previous realizations. This is achieved by systematically shifting actual electricity demand for the subsequent time period into the present period's target prediction, effectively converting time series data into a row-wise supervised learning problem. Explanatory variables may also be lagged to establish correlations between the current timeframe and previous periods.

Encoding

In decision trees, selecting a continuous variable for splitting can result in growth in both directions due to the multiple possible splitting values. However, categorical variables, particularly those with few levels such as one-hot encoded variables, are limited in their splitting options, resulting in sparse decision trees that grow in one direction towards zeroes in dummy variables. As dummy variables are treated as independent, making a split on one has little impact on the gain in purity per split, making it unlikely for the tree to select a dummy variable closer to the root. Checking feature importance can reveal the effectiveness of this method. One-hot encoding also obscures the order of importance of features not involved in the encoding, leading to inefficient models. [25]

Training and validation

In machine learning, data partitioning involves the separation of data into three categories: training, validation, and testing. K-fold cross-validation is often employed to improve model accuracy and reliability. However, time series data presents a unique challenge due to the temporal nature of observations, which makes random partitioning of data impossible. Instead, the data must be split in temporal order.

Backtesting is a crucial step in evaluating the accuracy and reliability of models in capturing underlying patterns and trends in time series data. In this study, the TimeSeriesSplit module was employed in combination with GridSearchCV (discussed below) to partition the provided dataset into training and validation sets. Notably, this approach allows for the use of the same two years of data for both training and validation, as well as for hyperparameter tuning, while reserving the final year of data for testing. This strategy ensures that the model's performance can be accurately evaluated on unseen data, and any potential issues with overfitting can be identified and addressed. [26]

Prediction interval

The out-of-bag (oob) error metrics method, as described by Andrew P. Wheeler in his article Prediction intervals for Random Forests [30], will be employed in this study to produce the prediction interval. The efficacy of this approach and its outcomes have been discussed in several studies, indicating that the proposed technique yields narrower intervals than other methods while maintaining marginal coverage rates that are approximately consistent with nominal levels. [27] [28]

Hyperparameters

In the context of hyperparameter optimization for random forests in classification analysis, a common approach is to employ GridSearchCV. Nevertheless, this approach is not always applicable for time series analysis since standard cross-validation methods cannot be employed. To address this issue, a possible alternative is to use the TimeSeriesSplit module, as illustrated in Dr. Varshita Sher's publication on time series modeling using scikit-learn, Pandas, and NumPy. The TimeSeriesSplit module allows for the partitioning of data into training and testing sets while preserving the chronological order of the data. Notably, successive training sets are supersets of those that come before them. Such a strategy ensures that the model is trained on data that is representative of the entire time series, making it particularly useful for time series analysis where temporal dependencies are a key consideration. By integrating TimeSeriesSplit with GridSearchCV, it is possible to optimize the hyperparameters of random forests specifically for time series analysis. [29]

Random Forest results

After conducting a grid search with an interval ranging from 100 to 1000 estimators and max depth ranging from 5 to 10, it was determined that the optimal model was achieved with 500 estimators and a max depth of 10. This outcome suggests that the selected parameter values can generate a highly efficient model that accurately captures the patterns and fluctuations of energy demand.

This efficiency is depicted in the plot below, the CMAPE exhibits an initial spike to 7%, followed by a subsequent decline to approximately 5%. This observation suggests that there may be an underlying factor that initially impacted the accuracy of the model but was subsequently mitigated.

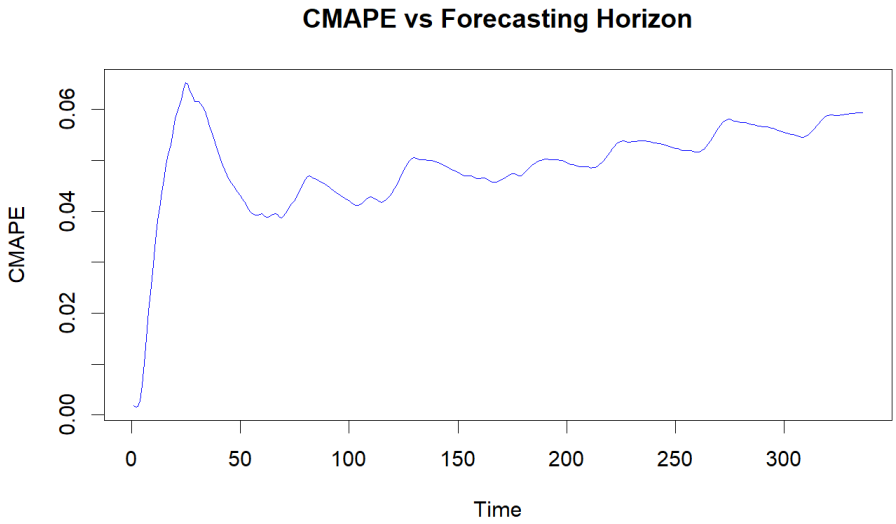


Fig. 15. Cumulative MAPE of random forest forecast for the first week of August 2021.

The feature importance plot below illustrates that the most significant features were time-related, specifically time weekday and month, as well as lagged differences in demand. This observation suggests that temporal patterns and shifts in demand are critical factors to consider when analysing the data, and underscores the importance of incorporating time-related features into predictive models.

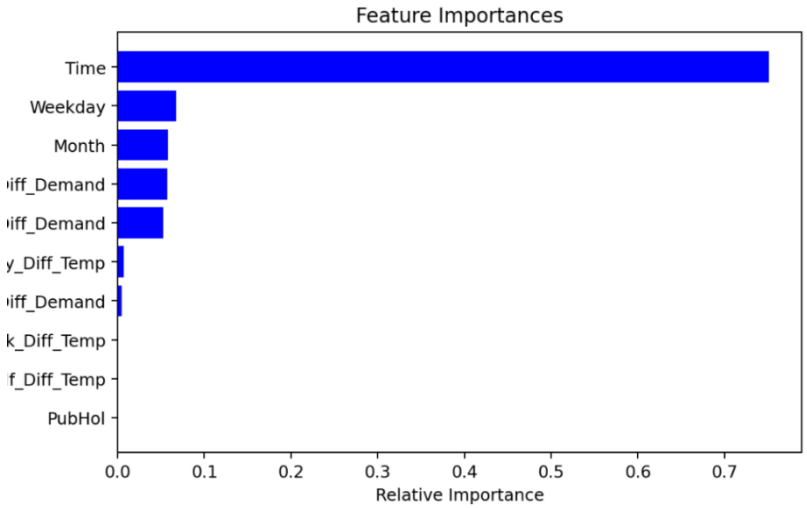


Fig. 16. Feature importances from random forest.

From the plot of forecast and prediction interval below, it is evident that the predicted values closely align with the actuals from the previous two weeks, and that the 90% prediction interval is relatively narrow. This indicates that given the current prediction, 90% of all other predictions are expected to fall within this narrow range, signifying minimal variance and a high degree of confidence in the accuracy of the prediction. These findings suggest that the model is effectively capturing the underlying patterns and trends in the data, and may be reliable for making future predictions with a similar level of accuracy.

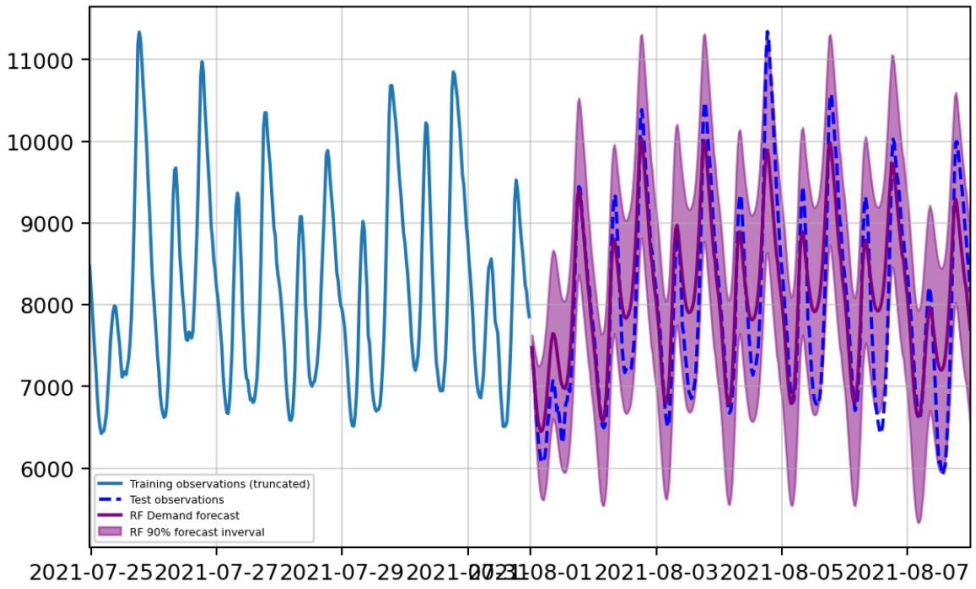


Fig. 17. Forecast and Prediction interval

Upon examining the Out of Bag (OOB) derived 90% prediction interval, we observed that the actual coverage rate is approximately 86%, which is considered sufficient for our purposes. This finding is consistent with the research conducted by H. Zhang et al. in their study [28], which suggests that prediction intervals tend to be slightly narrower and closer to the nominal coverage rate. The relevance of this finding is significant, as using OOB samples can help to improve the accuracy of the prediction interval by providing a measure of how well the model is performing on new, unseen data. Additionally, having a coverage rate close to the nominal rate indicates that the prediction interval accurately captures the uncertainty inherent in the data and can provide valuable information for making informed decisions based on the predicted outcome. Therefore, our analysis confirms the validity of the prediction interval and supports the reliability of the model in generating accurate predictions with a high degree of confidence.

Description: df [336 × 1]

	colMeans(cover) <dbl>
T+1	0.8734943
T+2	0.8687227
T+3	0.8723305
T+4	0.8739017
T+5	0.8723305
T+6	0.8676171
T+7	0.8615071
T+8	0.8549316
T+9	0.8517894
T+10	0.8481234

1-10 of 336 rows

Description: df [336 × 1]

	colMeans(cover) <dbl>
T+291	0.8792552
T+292	0.8774513
T+293	0.8753564
T+294	0.8705848
T+295	0.8679662
T+296	0.8663951
T+297	0.8599942
T+298	0.8565028
T+299	0.8528368
T+300	0.8500436

291-300 of 336 rows

Description: df [336 × 1]

	colMeans(cover) <dbl>
T+331	0.8640675
T+332	0.8634856
T+333	0.8651149
T+334	0.8659878
T+335	0.8677335
T+336	0.8680244

331-336 of 336 rows

v. LSTM

Neural networks have progress substantially in recent years, and it has been applied into broader areas including time series forecast. It has demonstrated as a powerful tool as they can study and learn complex patterns embedded in the data. In the energy sector, it has become very common practice to use neural networks to predict the electricity demand.

There are several different types of neural networks, and we select LSTM to tackle our problem for a couple of reasons. First, time series is a typical type of sequential data, which means the data come in order. Recurrent neural network (RNN) is a particularly designed for this type of data as it tries to remember the order or sequence of the data, and perform the prediction based on it. Among the various variants of RNN, Long Short-Term Memory (LSTM) is well known for its reliable performance handling long sequence.

Challenges

Our goal is to predict demand for 7 days at 30 mins interval, which is 336 timesteps. It is a very long series, so that create some challenges in terms of modelling.

There are two ways to produce multi-steps forecast, first is to direct forecast, and second approach is recursive forecast. [31]

- Direct forecast. The challenge of applying direct forecast in our case is the size of input and output, if we want to directly forecast 336 timesteps, it will require a larger size of input. E.g. if we want to know what would happen in next timestep, probably we only need to know what has happened in the 5 timesteps before, however, if we want to forecast next 10 timesteps, it would make more sense to make the forecast based on what happened in the previous 50 rather than 5 timesteps, so we can imagine how much each input size would be if we want to produce 336 output directly. To accommodate the large size of input and output, we may have to increase the complexity of the model substantially, and meanwhile, the time and complexity of training the model would also increase accordingly.

- Recursive forecast. This approach only forecast one timestep at a time and use the new forecast as part of the input to forecast one more step, and this process recurs until we get enough timesteps that we want. The model behind would be much less complex and easy to build. However, the issue associated with it is that we build forecast on forecast, and each forecast undoubtedly has some level of error, which will keep building up and getting bigger and bigger as the number of timesteps increase.

The approach we take is a balance between the two, which means the model is trained to produce small multi-steps, and then we recursive forecast to extend the forecast. Because the model would produce multi-steps, the required number of recursions would be reduced a lot, and we think that is the sweet spot between complexity and accuracy. We tried different size of input and output and experiment the model setup and find that 6 steps direct forecast based on 336 input is a comfortable size, and we then use as it as part of the input for next forecast, it repeats 56 times to get 7 days forecast window we want.

Data preparation

Preparing the data for the model training is critical. The first issue we need to fix is the scale of the data. In our dataset, demand can vary between 7000 and 11000, while the temperature only change within small two digits and it would create problems for the model to be trained. We applied standardization which scales each variable by subtracting its mean and dividing by its standard deviation and eventually all variables are brought to the same scale with a mean of zero and standard deviation of one.

After that, we need to reshape the data for the model training. We create sliding window of the training data with a step of 6. The first sample X is timestep 0 to 336, and Y is timestep 336 to 342, the second sample X is timestep 342 to 678 and Y is 678 to 684, and so on, as illustrated in Fig. 18. There are several benefits of adding the steps. Firstly, it reduces the size of the training data by 6 times compared to no step, which resulting a faster training process as well as preventing overfitting. Secondly, the step size is the same as the forecast size, so when we connect the forecast together, it is a continuous series without any further processing.

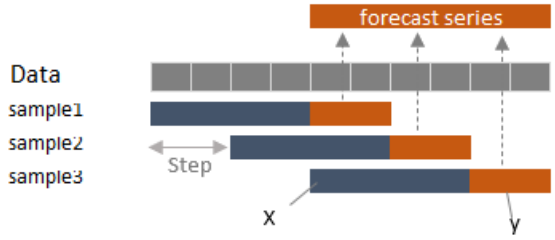


Fig. 18. Illustration of the sliding window with steps.

Model setup

Now we start to build our LSTM model, we used sklearn package in python for the model building as it is very easy to use. We setup double LSTM layer model to increase the depth of the model to capture more complex information [32]. We also add a dropout layer to prevent outfit, followed by an output layer with 6 units to match the number of forecast steps. The summary of the model is shown in Fig. 19. There are 124 thousand parameters in the model in Fig. 20, which should be enough to handle the complexity of the training data.

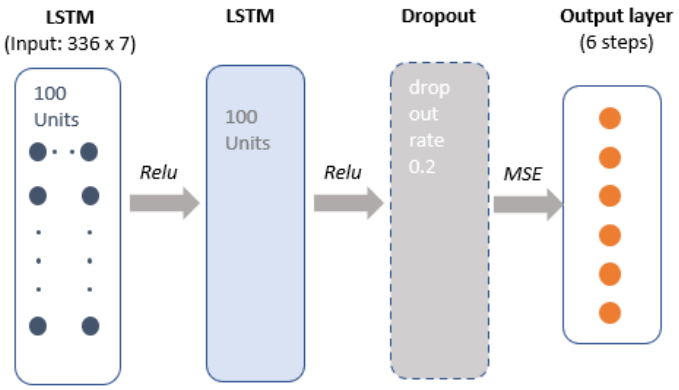
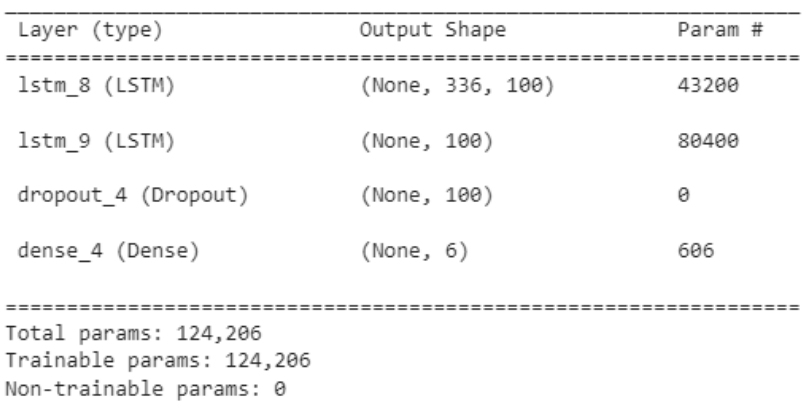


Fig. 19. Illustration of the LSTM layers.



Layer (type)	Output Shape	Param #
lstm_8 (LSTM)	(None, 336, 100)	43200
lstm_9 (LSTM)	(None, 100)	80400
dropout_4 (Dropout)	(None, 100)	0
dense_4 (Dense)	(None, 6)	606

Total params: 124,206
Trainable params: 124,206
Non-trainable params: 0

Fig. 20. Parameters in the LSTM model.

- Activation function defines the output of each layer in the networks, and we use rectified linear (Relu) as it can overcome the vanishing gradient problem that is commonly seen in other functions like sigmoid and hyperbolic tangent, and it allows models to learn faster [33].

- Loss function measures how well the model performs against the targets/labels, and we used mean squared error (MSE) in this model.

- Optimizer is an algorithm that modify the parameters like weights and learning rates to reduce the loss we defined. We used Adaptive Moment Estimation (Adam) in our model. Adam combines the strength of other algorithms like Adagrad and RMS prop, and it will find an adaptive rate during the training process, and it is easy to implement without too much tunning and fast to run. Adam is adapted as a benchmark for deep learning pater and recommended as a default optimization algorithm. [34].

- Learning rate specifies how fast the model leans. Normally we can use the default learning rate in Adam, but in our model, we had to assign a smaller learning rate (smaller than 0.0001) to avoid gradient explosion which leads to nan as the training result.

Performance

After being trained with the train set, the model is capable of 3 hours forecast, we used it to forecast first 7 days in our test set.

Both actual demand and forecast were plotted together as in Fig. 21. Although it tends to underestimate the peak demand at early mornings and early evenings, the overall accuracy is

very high with only 3.4% MAPE, and the accuracy is consistent at most timestep.

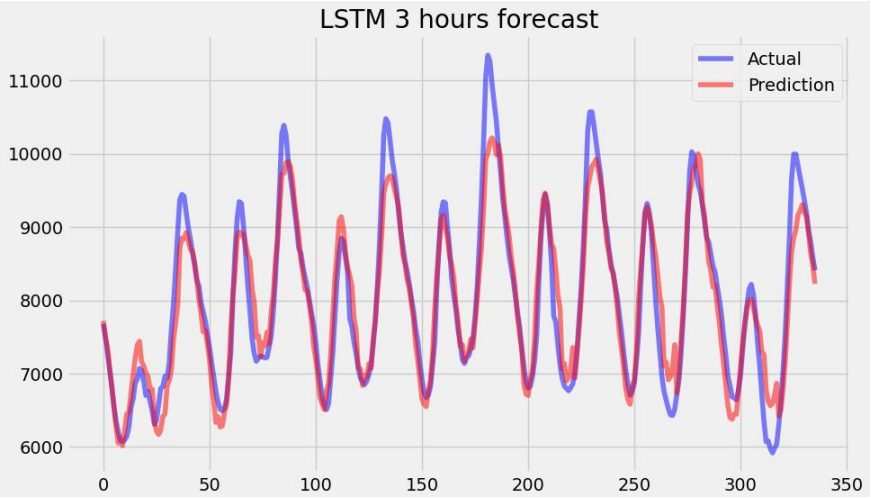


Fig. 21. LSTM 3-hour forecast for the first week of August 2021.

Next, we applied the recursive approach to extend the forecast length to 7 days, and the results were shown in Fig. 22. As we can see, the forecast is pretty good at the beginning, as the forecast window moves further, the accuracy gradually decreased. It is not surprising as the proportion of actual in the input data became less and less in later recursion. In other words, as we forecast based on forecast, the error accumulates, which is illustrated in Fig. 23. However, the overall accuracy was still very good, with a MAPE of under 5% and it is way better than our baseline linear regression model.

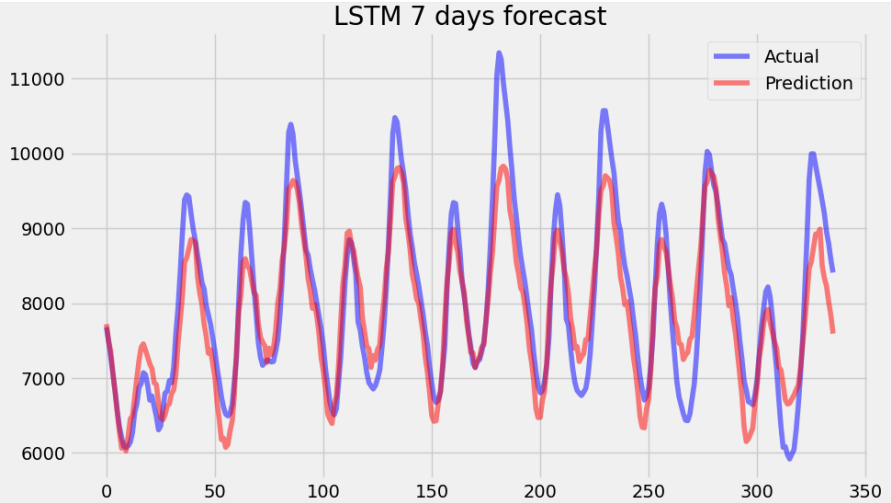


Fig. 22. LSTM 7-day forecast for the first week of August 2021.

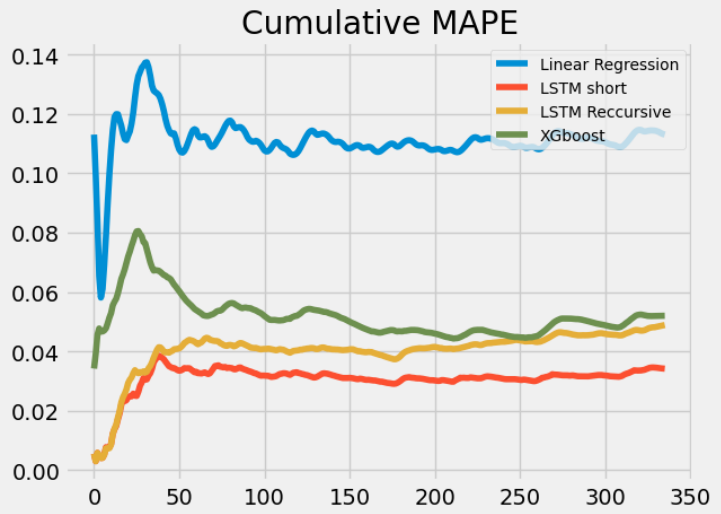


Fig. 23. Cumulative MAPE comparison.

vi. XGboost

Another machine learning approached we experimented is Extreme Gradient Boosting (XGBoost). It is an ensemble method that make predictions based on multiple decision trees, and it can be applied to solve a wide range of classification and regression problems. XGBoost employs stochastic gradient boosting machine learning algorithm and is well known for its efficiency and scalability [35]. When we apply it to our forecast, we did find it very easy and fast to train and perform a decent forecast without too much tuning. We followed similar approach as the other model, mainly rely on the temperature and time features as variable, and on top of that, we added some four lag features, which are the demand from the 4 timesteps 7 days right before.

We then trained the model with 2000 estimators and used it to predict the last 7 days, and the prediction result is plotted in Fig. 24. Overall, it is pretty good with a MAPE around 5%, and interestingly, opposite to LSTM where we see most underestimates for the peak period, XGBoost consistently overestimates the demand in the mid-day low demand period.

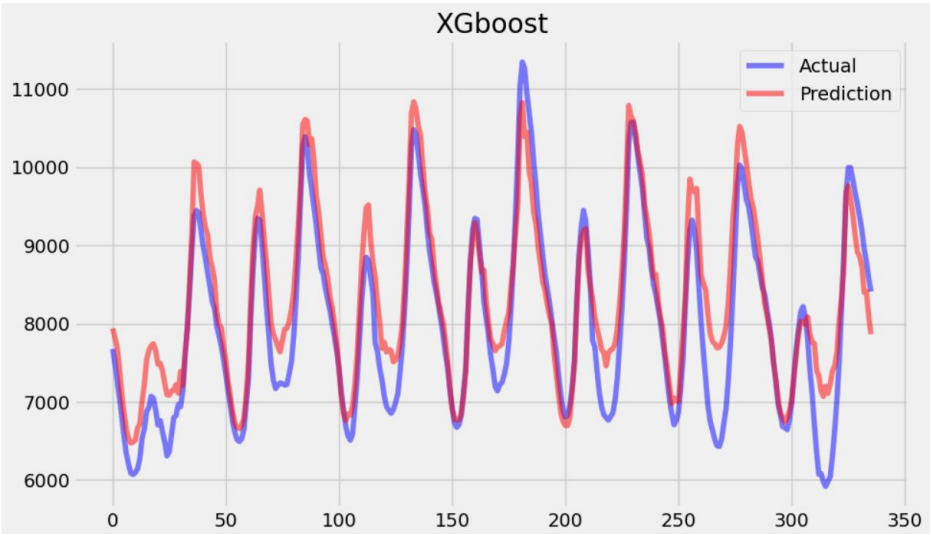


Fig. 24. XGBoost forecast for the first week of August 2021.

Another interesting feature from regression methods is the feature importance, which is basically the coefficients. A big positive or negative efficient means the variable have a larger influence in the whole equation and vice versa. The feature importance from XGBoost is plotted in Fig. 25. It suggests that the lag1 which is the demand 7 days ago is the most relevant feature, followed by season. Day type (weekday, weekend, or holiday) and Half hour index come after with similar importance.

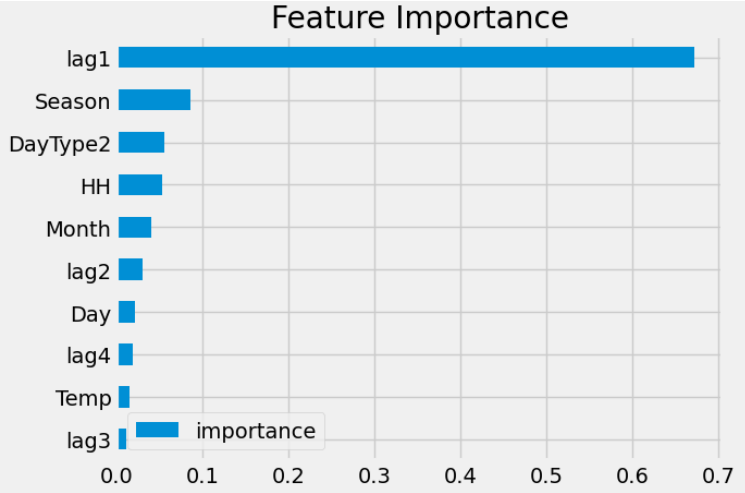


Fig. 25. Feature importance suggested by XGBoost.

6. *Consolidation and Comparison:*

i. Performance for the first weeks of test set.

We consolidate all the forecast, measured the accuracy, and the performance of the first weeks of the test set is presented in Fig. 26. All models beat the baseline linear regression model, and most models' performances are fluctuating in a range between 5% to 6% and LSTM is the most accurate model for week one forecast.

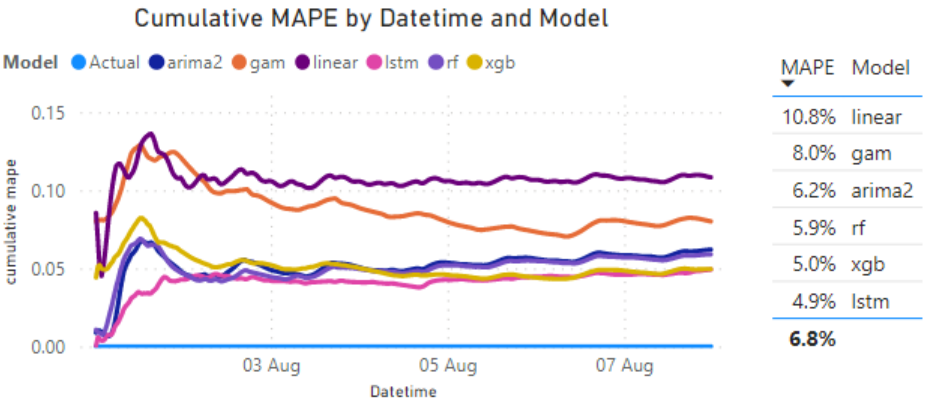


Fig. 26. MAPE of different models for the first week of August 2021.

Further study based on 52 weeks forecast.

But one weeks is a short period from just one month in one season and may not be representative enough. Before we deploy any of the model for demand forecast in real environment, we need to examine how each model perform in as many scenarios as possible. In that perspective, we zoom out a bit and compare the performance over a longer horizon – the one-year test set that we reserved, which covers 52 weeks forecast cycle. For linear regression model and Arima, we just simply use the model predict demand based on the variables (temperature and time features) in the test set. For other models, as they have used lag features as the input, the one-year forecast is essentially a combination of 52 individual forecast done at the beginning of each week of the test period.

We built a dashboard in power BI with all the forecast and its residual, so we can easily slice and dice the result from different perspectives to see if any patterns.

ii. Seasonal Pattern

When we look at the MAPE of each model over different months, there is an interesting pattern that we observe in Fig. 27.

All the statistical based models seem stable over the year without too many fluctuations. However, on the machine learning side. Random forest (dark blue line) performs well from January to July, but its forecast accuracy starts to decrease in the second half of the year, while XGBoost (pink line) is doing the opposite, where the performance is not so great in the first half but shows one of the best performances in later of the year. This pattern is also clearly supported in the weekly trend in Fig. 28 (our test set starts in August, which the week 1 is beginning of august).

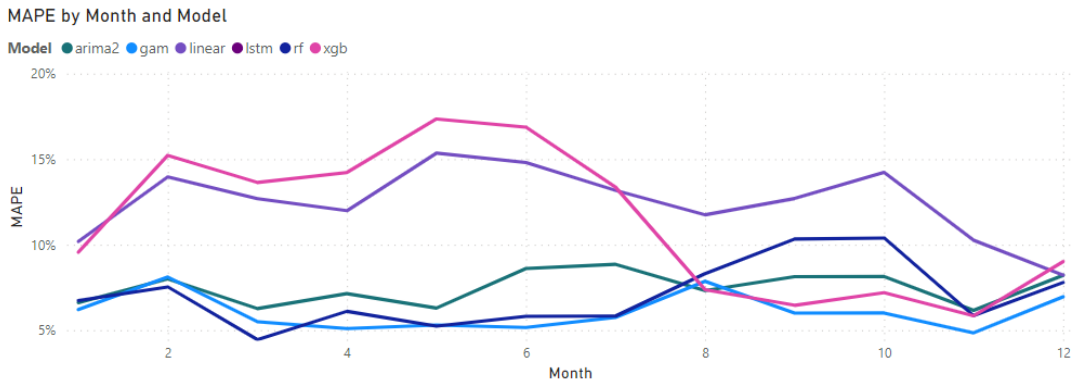


Fig. 27. MAPE of different models by month.

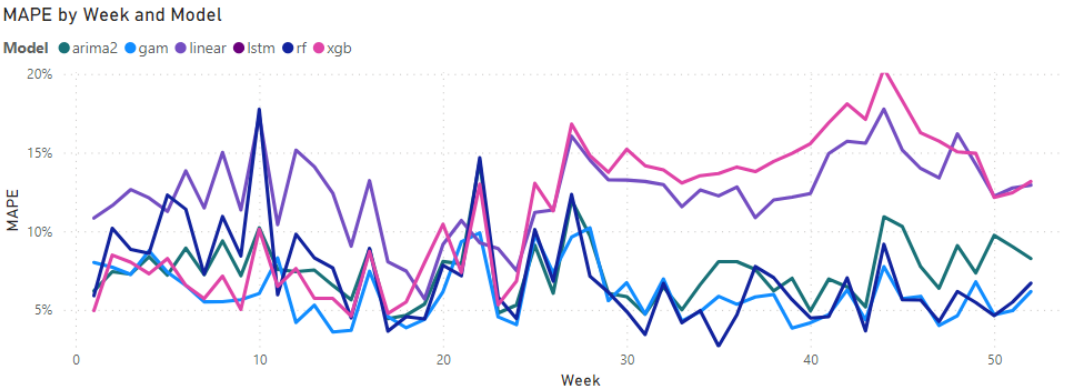


Fig. 28. MAPE of different models by week.

iii. LSTM explosion

We may have noticed that LSTM, which has the best accuracy mentioned earlier, is not in the monthly and weekly trend. This leads us to further investigate and have another interesting finding.

From the LSTM section, due to the model complexity and resource constraints, we applied recursive approach to extend 3 hours direct forecast to 7 days forecast. One of the key risks here is the accumulative error, which can be dangerous. In our case, we see the residual become off-track in the mid-June not long after a new cycle of 7-day forecast. The error keeps building on, and the forecast goes wild until the next forecast cycle starts on 19th of June. The residual exploded so much that we can only show it together with normal residuals on a log scale in Fig. 29. It might be triggered by extreme weather change, and we only see this kind of explosion once over the whole year forecast. However, if it happens in the real business environment and the data being used without diligent check, the cost could be significant.

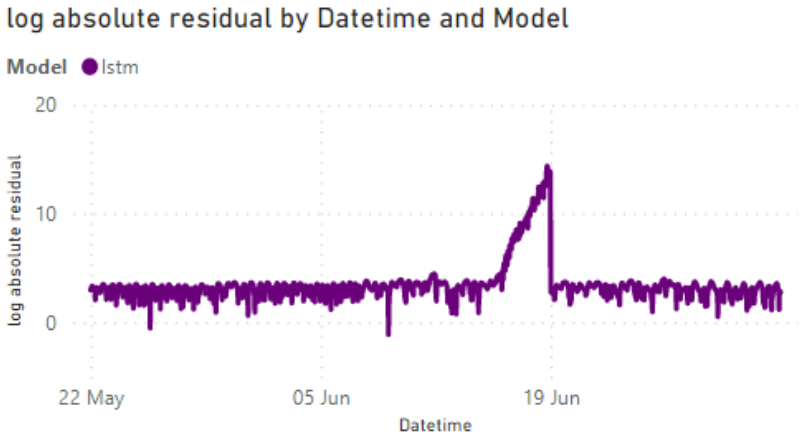


Fig. 29. LSTM residual explosion in the week 46 forecast.

Based on what we see above, we can conclude that having only one model is not enough to provide consistent and most accurate demand forecast as each model has its strengths and weaknesses. A better practice maybe having multiple models running parallel. I.e., Using statistical model can provide some more reliable base forecast for checking and long-term forecast, and machine learning models to get a more accurate short-term forecast, and if possible, using a couple of different machine learning models to be used in different scenarios like season, month etc to get the best results.

iv. Pros and Cons

In this project, we learnt that each model has its strength and weakness. It is recommended that businesses choose models based on their own requirements and resources. To help make that decision, we have summarized the pros and cons are summarized in the table below.

Model	Pros	Cons
Linear Regression	• Very easy to implement. • Reliable for long term forecast.	• Accuracy is not very well. • Not able to capture complex trend.

ARIMA	• Simplicity: ARIMA models are straightforward as they use simple components like autoregressive (AR), moving average (MA), and integrated (I) parts. • Flexibility: ARIMA models can handle different patterns in time series data, such as trends and seasonal changes, by adjusting the AR, I, and MA components. • Wide Applicability: ARIMA models can be used for many types of time series data, like stock market prices, economic data, and weather observations. • Forecasting: ARIMA models can make reliable short-term predictions based on past patterns and connections in the data. • Diagnostic Tools: ARIMA models provide helpful tools like autocorrelation and partial autocorrelation plots that can assist to choose the best model and check how well it fits the data.	• Stationarity Assumption: ARIMA models require the time series data to be stationary or differenced to be stationary. That is not applicable to all data. • Limited Multivariate Analysis: ARIMA models mostly deal with one variable at a time and may not capture the relationships between multiple variables well. • Linear Relationships: ARIMA models do not work well for complex or non-linear pattern as they focus on simple, linear relationships between past and present values. • Sensitivity to Model Order: ARIMA models can be affected by the settings (p, d, q), and identifying the best settings requires trial and error or automatic selection methods. • Lack of External Factors: ARIMA models' prediction results may be affected by the model feature that they do not consider the impact of outside factors on the time series by default. This issue can be solved by ARIMAX or SARIMAX models that include these factors.
Random Forecast	• Non-parametric and can handle linear and non-linear relationships well. • Easy to understand and performance is less susceptible to outliers. • Can be used for both categorical and regression tasks. • Implicitly perform feature selection and generate uncorrelated decision trees. • Generally, provide high accuracy and balance the bias-variance trade-off well	• Needs adjustment for time series extrapolation. • Provides feature importance but no visibility into the coefficients like linear regression. • Computationally intensive for large datasets
GAM	• Easy to implement. • Flexible to include many variables. • Good for long term forecast.	• Accuracy is not the best.
LSTM	• Top accuracy for short-term forecast. • No assumptions need to be made.	• Need a lot of resources to properly build and train the model. • "Black box" - hard to explain the result or mechanism. • Difficult to forecast long time series.
XGBoost	• Very easy to train and implement. • Good Scalability, flexible to extend to long term forecast	• The performance seems not very stable in some seasons.

v. Hybrid Model Experiment

From the models that we have explored so far, they all have different strength and weakness. It makes us think if it is possible to combine the statistical based model and machine learning together to improve the overall accuracy. Here is our experiment using GAM + XGBoost with the procedures below in Fig. 30:

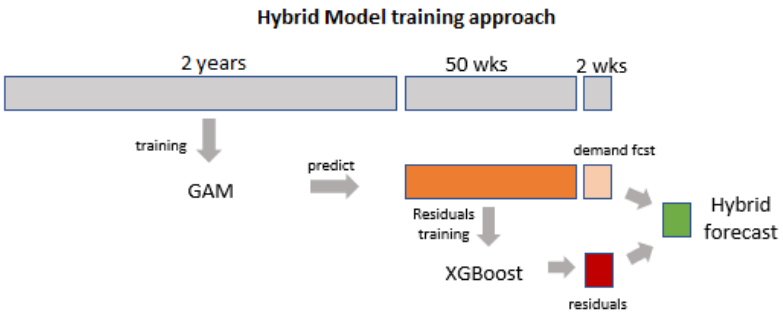


Fig. 30. Illustration of the Hydrid model approach.

We trained the GAM model and use it to predict the one-year test data, then we calculate the residuals of the forecast at each time steps. Then we reserve the last two weeks as test data and using the remaining residuals as well as other time features to train the XGBoost model. The idea is that hoping XGBoost can identify some patterns of the residuals (what has been missing by the statistical model) and find a way to compensate. We then use the trained XGBoost model to predict the residual for the reserved one-week data, and we add this prediction to the GAM forecast as our final forecast.

The comparison is presented in Fig. 31. We can see that the XGBoos indeed identified the consistent underestimation for the peak hours from GAM model and tried to compensate the forecast a little bit for those periods, and the overall MAPE is consistently lower than the GAM model alone as shown in Fig. 32. Especially the first week the MAPE is about 1% better, which is a 14% improvement on the accuracy, and it can make a significant difference in terms of optimizing the bidding strategy for the generators.

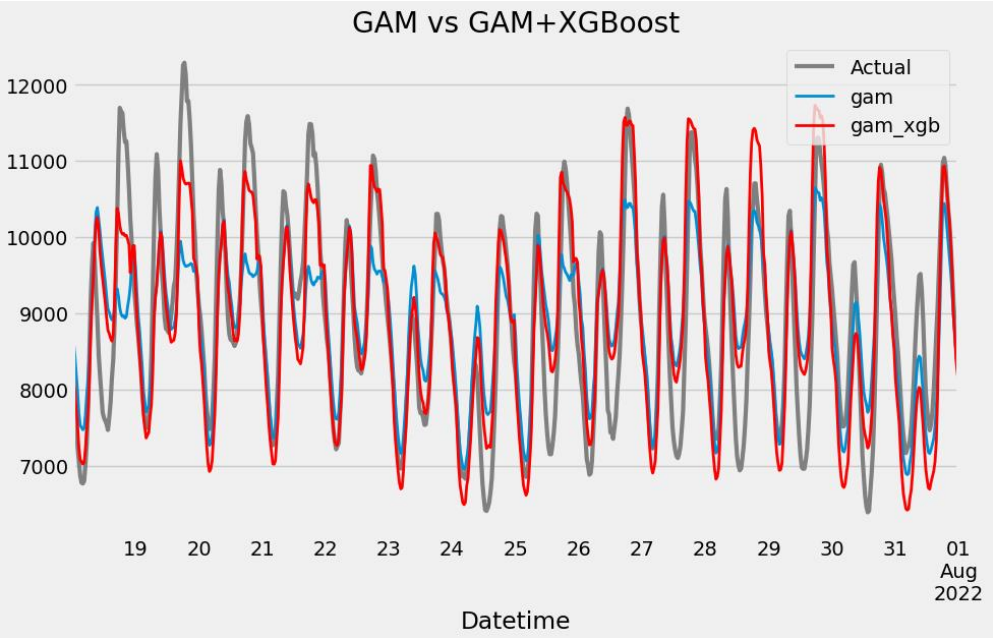


Fig. 31. Forecast accuracy comparison between GAM model and GAM + XGBoost model.

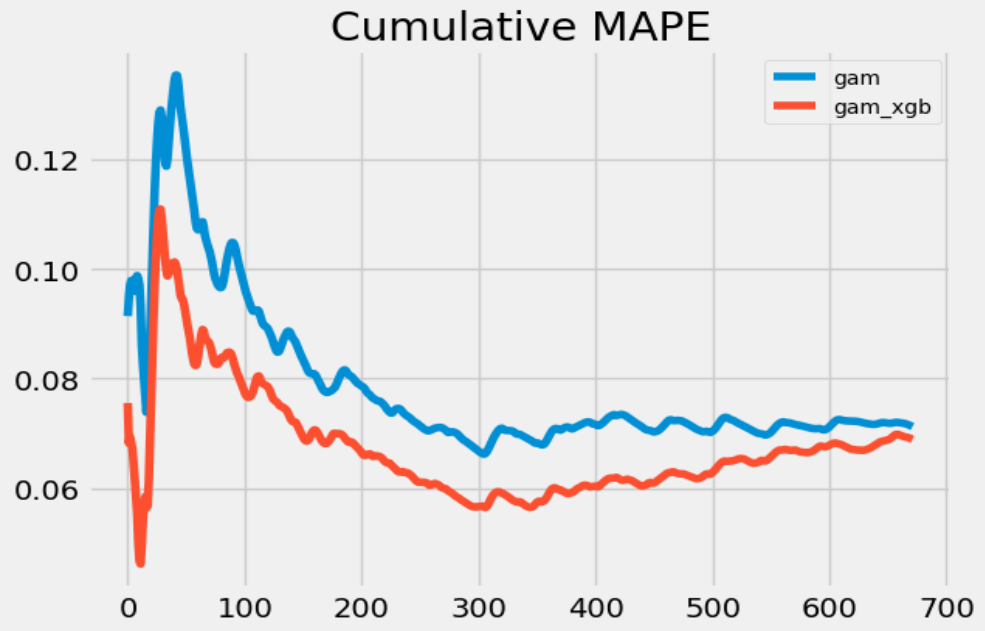


Fig. 32. Cumulative MAPE comparison between GAM model and GAM + XGBoost model.

7. Further Discussion

i. Model Switching

The presence of multiple models forecasting electricity demand results in a range of demand forecasts that can be ranked by their overall predictive performance. However, as the predictive performance of each model can vary over time due to natural fluctuations in energy consumption, its temporal performance may significantly differ from its overall performance. The concept of "Model Switching" leverages these temporal variations to generate a single optimized forecast for energy market participants, maximizing forecasting accuracy based on a predefined time horizon. Model Switching can be achieved in two ways:

Minimum variance

The method of selecting the most precise prediction within a recent period based on the minimum residual, provides an advantage in situations where all independent models over or underestimate true demand in consecutive periods. In such scenarios, the method of selecting the best performing forecast within a recent time frame enables greater accuracy in forecasting future demand values.

Averaging

Empirically weighted averaging can outperform minimum variance methods in cases where the actual demand falls between the forecasts of the two closest models. This is because minimum variance methods focus on selecting the most accurate forecast, which may not necessarily consider the relative accuracy of other forecasts. In contrast, empirical weighted averaging involves assigning weights to the forecasts based on their past performance and using these weights to compute a weighted average forecast. This approach can lead to

improved accuracy in cases where multiple models are providing forecasts that are close to the actual demand value.

ii. Composite Forecast

In situations where multiple time horizons are required, the Model Switching technique may result in different optimal models for different horizons. For instance, Model A may be optimal for a 1-day horizon while Model B may be optimal for a 7-day horizon. To generate a forecast that incorporates the strengths of both models, a composite forecast can be created. This involves using the forecast of Model A for the first day and the forecast of Model B for the remaining 6 days to produce a single, more accurate forecast for the entire 7-day horizon.

8. Reference

1. AEMO, "Load forecasting in pre-dispatch and STPASA" in aemo.com.au, 2022. Accessed: Mar. 10, 2023. [online]. Available: https://aemo.com.au/-/media/files/electricity/nem/security_and_reliability/dispatch/spot-market-operations-timetable.pdf

2. Al-Alawi, S. M., & Islam, S. M. (1996). Principles of electricity demand forecasting. I. methodologies. *Power Engineering Journal*, 10 (3), 139–143.

3. Arora, S., & Taylor, J. W. (2013). Short-term forecasting of anomalous load using rule-based triple seasonal methods. *IEEE Transactions on Power Systems*, 28 (3), 3235–3242.

4. Lee CC, Chiu YB (2011) Electricity demand elasticities and temperature: evidence from panel smooth transition regression with instrumental variable approach. *Energy Econ* 33:896–902.

5. Xiao N., Zarnikau J., Damien P. (2007) Testing functional forms in energy modeling: an application of the Bayesian approach to the US electricity demand. *Energy Economics*, 29 (2007), pp. 158-166.

6. Henley A, Peirson J (1997) Non-linearities in electricity demand and temperature: parametric versus non-parametric methods. *Oxford Bull Econ Stat* 59:149–162.

7. De Felice M, Alessandri A, Ruti PM (2013) Electricity demand forecasting over Italy: potential benefits using numerical weather prediction models. *Electr Power Syst Res* 104:71–79.

8. Apadula F, Bassini A, Elli A, Scapin S (2012) Relationships between meteorological variables and monthly electricity demand. *Appl Energy* 98:346–356.

9. Marvuglia A, Messineo A (2012) Using recurrent artificial neural networks to forecast household electricity consumption. *Energy Procedia* 14:45–55.

10. Bessec M, Fouquau J (2008) The non-linear link between electricity consumption and temperature in Europe: a threshold panel approach. *Energy Econ* 30:2705–2721.

11. Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and Practice (3rd ed.)*. Monash University, Australia. Retrieved from *https://otexts.com/fpp3/*

12. Wang Y, Bielicki JM. Acclimation and the response of hourly electricity loads to meteorological variables. Energy. 2018 Jan 1;142:473-85. Available at: *https://www.sciencedirect.com/science/article/pii/S0360544217317061* (Accessed: 13 Mar 2023).

13. Simpson, J (2021) *How does AEMO predict demand in the National Electricity Market?* Available at: *https://jacksimpson.co/how-does-aemo-predict-demand-in-the-national-electricity-market* (Accessed: 12 Mar 2023).

14. Shin, S. Y., & Woo, H. G. (2022). *Energy Consumption Forecasting in Korea Using Machine Learning Algorithms. Energies. 15. 4880. 10.3390/en15134880.* Available at: *https://www.researchgate.net/publication/361717016_Energy_Consumption_Forecasting_in_Korea_Using_Machine_Learning_Algorithms*

15. Gavin L Simpson, "Modelling Palaeoecological Time Series Using Generalised Additive Models" in METHODS article, 2018. Accessed: Apr. 1, 2023. [online]. Available: https://www.frontiersin.org/articles/10.3389/fevo.2018.00149/full

16. Box, G. E. P and Jenkins, G.M., (1976). Time series analysis: Forecasting and control, *Holden-Day*.

17. Elamin, N., & Fukushige, M. (2018). Modeling and forecasting hourly electricity demand by SARIMAX with interactions. *Energy*, 165, 257–268.

18. Fan, S., & Chen, L. (2006). Short-term load forecasting based on an adaptive hybrid method. *IEEE Transactions on Power Systems*, 21 (1), 392–401.

19. Caro, E., Juan, J., & Cara, J. (2020). Periodically correlated models for short-term electricity load forecasting. *Applied Mathematics and Computation*, 364, 124642.

20. Dudek, G. A "Comprehensive Study of Random Forest for Short-Term Load Forecasting" Energies 2022, 15, 7547. Accessed: Mar 15, 2023 [online]. Available: https://doi.org/10.3390/en15207547

21. Hafidz Zulkifli, "Multivariate Time Series Forecasting Using Random Forest" in Towards Data Science. Accessed: Mar 25, 2023 [online]. Available: https://towardsdatascience.com/multivariate-time-series-forecasting-using-random-forest-2372f3ecbad1

22. Sarem Seitz, "Forecasting with Decision Trees and Random Forests" Accessed: Mar 20, 2023 [online]. Available: https://www.sarem-seitz.com/forecasting-with-decision-trees-and-random-forests/

23. Aman Arora, "Why Random Forests can't predict trends and how to overcome this problem?" in Medium. Accessed: Mar 15, 2023 [online]. Available: https://medium.datadriveninvestor.com/why-wont-time-series-data-and-random-forests-work-very-well-together-3c9f7b271631

24. Sckit-Learn documentation "Multi-output problems" Accessed: Mar 15, 2023 [online]. Available: https://scikit-learn.org/stable/modules/tree.html#multi-output-problems

25. Rakesh Ravi, “One-Hot Encoding is making your Tree-Based Ensembles worse, here’s why?” in Medium. Accessed: Mar 20, 2023 [online]. Available: https://towardsdatascience.com/one-hot-encoding-is-making-your-tree-based-ensembles-worse-heres-why-d64b282b5769

26. Jason Brownlee, “How To Backtest Machine Learning Models for Time Series Forecasting” in Machine Learning Mastery. Accessed: Mar 18, 2023 [online]. Available: https://machinelearningmastery.com/backtest-machine-learning-models-time-series-forecasting/

27. Roy M-H, Larocque D. “Prediction intervals with random forests.” In Statistical Methods in Medical Research. 2020;29(1):205-229. doi:10.1177/0962280219829885. Accessed: Mar 23, 2023 [online]. Available: https://journals.sagepub.com/doi/full/10.1177/0962280219829885

28. Haozhe Zhang, Joshua Zimmerman, Dan Nettleton & Daniel J. Nordman (2020) “Random Forest Prediction Intervals”, The American Statistician, 74:4, 392-406, DOI: 10.1080/00031305.2019.1585288 Accessed: Mar 23, 2023 [online]. Available: https://www.tandfonline.com/doi/full/10.1080/00031305.2019.1585288

29. Dr Varshita Sher, “Time Series Modeling using Scikit, Pandas, and Numpy” in Medium. Accessed: Mar 23, 2023 [online]. Available: https://towardsdatascience.com/time-series-modeling-using-scikit-pandas-and-numpy-682e3b8db8d1

30. Andrew P Wheeler, “Prediction Intervals for Random Forests” Accessed: Mar 23, 2023 [online]. Available: https://andrewpwheeler.com/2022/02/04/prediction-intervals-for-random-forests/

31. J. Brownlee, “4 Strategies for Multi-Step Time Series Forecasting” in Machine Learning Mastery, 2019. Accessed: Mar. 19, 2023. [online]. Available: https://machinelearningmastery.com/multi-step-time-series-forecasting/

32. J. Brownlee, “Stacked Long Short-Term Memory Networks” in Machine Learning Mastery, 2019. Accessed: Apr. 6, 2023. [online]. Available: https://machinelearningmastery.com/stacked-long-short-term-memory-networks/#:~:text=Stacking%20LSTM%20hidden%20layers%20makes,range%20of%20challenging%20prediction%20problems

33. J. Brownlee, “A Gentle Introduction to the Rectified Linear Unit (ReLU)” in Machine Learning Mastery, 2019. Accessed: Mar. 19, 2023. [online]. Available: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

34. Gupta A, “A Comprehensive Guide on Optimizers in Deep Learning” in Analytics Vidhya, 2021. Accessed: Mar. 20, 2023. [online]. Available: https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/#What_Are_Optimizers_in_Deep_Learning?

35. J. Brownlee, “How to Use XGBoost for Time Series Forecasting” in Machine Learning Mastery, 2020. Accessed: Apr. 3, 2023. [online]. Available: https://machinelearningmastery.com/xgboost-for-time-series-forecasting/

9. Appendix

The following code from their authors has contributed to the creation of some outputs in this report.

27 Sarem Seitz, “Forecasting with Decision Trees and Random Forests”

Generating Forecast and Prediction interval plot

```
model = RandomForestARModel(n_lags = 2, log_transform = True, first_differences = True, seasonal_differ
model.fit(df_train)

predictions_forest = model.sample_forecast(n_periods=len(df_test), n_samples=10000)


means_forest = np.mean(predictions_forest,1)
lowers_forest = np.quantile(predictions_forest,0.05,1)
uppers_forest = np.quantile(predictions_forest,0.95,1)

plt.figure(figsize = (18,7))

plt.grid(alpha=0.5)

plt.plot(df.iloc[-120:], label = "Training observations (truncated)")
plt.plot(df_test, color = "blue", label = "Out-of-sample observations", ls="dashed")

plt.plot(df_test.index,means_forest,color="purple", label = "RF mean forecast")

plt.fill_between(df_test.index, lowers_forest, uppers_forest, color="purple", alpha=0.5, label = "RF 9
plt.legend(fontsize=13)
plt.margins(x=0)
```

random_forest_forecast.py hosted with ❤ by GitHub view raw

34 Dr Varshita Sher, “Time Series Modeling using Scikit, Pandas, and Numpy” in Medium

Gridsearch

```
from sklearn.metrics import make_scorer

def rmse(actual, predict):

predict = np.array(predict)
    actual = np.array(actual)

distance = predict - actual

square_distance = distance ** 2

mean_square_distance = square_distance.mean()

score = np.sqrt(mean_square_distance)

return score

rmse_score = make_scorer(rmse, greater_is_better = False)
```

Feature importance

```
imp = best_model.feature_importances_
features = X_train_2o_solar.columns
indices = np.argsort(imp)

plt.title('Feature Importances')
plt.barh(range(len(indices)), imp[indices], color='b',
align='center')
plt.yticks(range(len(indices)), [features[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()
```

35 Andrew P Wheeler, “Prediction Intervals for Random Forests”

Prediction Intervals

```
# Generating the error distribution
resid = train[y] - regr.oob_prediction_
# 50% interval
lowq = resid.quantile(0.25)
higq = resid.quantile(0.75)
print((lowq,higq))
# negative much larger
# so tends to overpredict time
```

```
# Generating predictions on out of sample data
test_y = regr.predict(test[x])
lowt = (test_y + lowq).clip(0) #cant have negative numbers
higt = (test_y + higq)

cover = (test[y] >= lowt) & (test[y] <= higt)
print(cover.mean())
```