A Data Science Approach to Forecast Electricity Consumption in Australia

Team Members

Student Name	**Student ID**
Joshua Rizk	z5571253
Josh Phillips	z5561466
Rachel Achampong	z5453628
Fuad Abo Dayyak	z5515774

Date: 24/04/2025

Table of Contents

Abstract

Accurate consumption forecasting is essential in optimising grid operations, integrating renewable energy sources, reducing costs, and informing decision-making for a range of stakeholders—including energy providers (e.g., AGL, Origin), the Australian Energy Market Operator (AEMO), policymakers, and consumers.

This report explores a data science approach to forecasting electricity demand, with the intent to identify the most accurate predictive method and assess whether temperature is the most influential variable in determining electricity demand.

First, a literature review explores traditional time series models like Holt-Winters and SARIMA, and contrasts them with advanced machine learning methods, particularly Long Short-Term Memory (LSTM) networks. Key challenges in modelling electricity consumption are addressed through feature engineering, such as seasonality, volatility, and the influence of exogenous factors like weather and holidays.

Using five-minute interval data from the Bankstown area, the modelling evaluates three predictive models explored in the literature review: Holt-Winters, SARIMA, and LSTM. The dataset includes historical demand, forecast demand, and temperature, spanning from 2010 to 2020.

The Holt-Winters model, limited by its inability to include external variables, performed poorly with a mean absolute percentage error (MAPE) of 15.5%. SARIMA, which incorporated temperature and calendar variables, achieved a significantly better MAPE of 5.2%. However, the best results were obtained with LSTM networks. A 48-hour input window LSTM model achieved the best balance of performance and efficiency, producing a MAPE of 3.23% and an R^2 of 0.92.

Although the 72-hour windowed LSTM achieved a marginally lower MAPE of 3.22%, the 48-hour model was preferred due to its better computational efficiency, smoother learning curve, and comparable accuracy. The 48-hour model displayed lower training complexity, faster convergence, and smaller gaps between training and validation loss, indicating stronger generalisation and practical suitability.

The findings confirm that temperature is a strong predictor of electricity demand, particularly when squared to capture non-linear effects, but it is most effective when used alongside engineered features such as lagged demand and time-based indicators. The report concludes that LSTM offers the best balance of predictive accuracy and scalability, recommending it as the preferred approach for stakeholders seeking high-performing, real-time forecasting solutions.

1. Introduction

Electricity consumption is crucial for all facets of life, as well as forming a major expense for households, businesses and government. As a result, short-term electricity forecasting is critical for an efficient grid operation, enabling better load planning and implementation of renewable energy sources. With the introduction of decarbonization initiatives, the need for short-term consumption forecasting is expected to grow exponentially (Sharifhosseini, S. M., Niknam, T., Taabodi, M. H., Aghajari, H. A., Sheybani, E., Javidi, G., & Pourbehzadi, M. 2024).

Electricity prices are difficult to forecast due to their inherent statistical properties – they tend to be highly volatile, non-linear, and non-stationary (Zhang, Wang and Wang, 2017). Despite these challenges it is important for a variety of stakeholders to be able to accurately predict short-term energy requirements. Effective short-term forecasting is needed to integrate renewable energy sources, such as wind and solar power, into the grid (Arslan Tuncar, E., Sağlam, Ş., & Oral, B. 2024). Accurate demand predictions further enhance grid stability, optimizing energy storage (Brailey 2024).

From a business perspective, having significant forecast errors can be the difference between a profitable operation, promoting the supply of more electricity leading to lower prices for consumers, and not. The benefits for producers of energy – AGL and Origin for example – are clear, but there are also numerous benefits for consumers. By having accurate short term energy forecasting, the Australia Energy Market Operator (AEMO), can maximize the use of lower cost producers, helping to keep electricity prices down (Australian Renewable Energy Agency 2019). Given the increasing government and societal focus in Australia on the cost of living, helping to lower electricity prices has a broad appeal.

For our analysis, historical demand for the Bankstown area has been sourced from AEMO. The data has a frequency of 5 minutes, which represent the five-minute dispatch cycles used to maintain grid stability (Zhang, R., Dong, Z. Y., Xu, Y., Meng, K., & Wong, K. P. 2012). Additionally, AEMO forecast demand and temperature data have been sourced to augment historical demand. A one hour forecast interval has been chosen. This interval is easier conceptually and appeals to a variety of stakeholders, an hour also tends to be the shortest forecasting period seen in the literature.

This report hence will suggest methods and models to accurately forecast hourly electricity consumption within Australia. This should appeal and be of use to a variety of participants within the Australian energy markets.

2. Literature Review

2.1 Traditional Time Series Models

The field of time series forecasting has progressed significantly, with increasingly complex computational methods in the last twenty years. Given the nature of the data, earlier forecasting studies focused on time series analysis – such as Holt Winters and Auto Regressive Integrated Moving Average (ARIMA) models. Given that Holt Winters models can decompose the seasonality of trend of time series behavior, they are aptly suited to electricity consumption. Taylor (2003) notes Holt Winters popularity, however, stipulates that it can only account for one seasonal trend. In his 2010 paper, Taylor (2010) expands upon this, after noting the success of exponential weighting methods on short time periods. By accounting for multiple seasonality's, an adjusted Holt Winters method was more effective than ARIMAs for short-term forecasts. Furthermore, Holt Winters is a simpler and more interpretable method, making it more practical for business use.

However, there are certain drawbacks to the Holt Winters model. For example, Waheed and Qingshen (2024) utilize hourly electricity consumption, and find that the Prophet Model outperforms Holt Winters on, and is more resistant to outliers. The Prophet Model, developed by Facebook, similarly uses a time-series decomposition, but can incorporate events into the analysis - such as holidays. It is then able to fit a linear or non-linear trend, while Holt Winters is only able to account for a linear trend. Considering Waheed and Qingshens' (2024) greater success with the Prophet Model, it may be chosen for our predictor. This study at least highlights the non-linear nature of electricity consumption.

SARIMA models have also been employed. SARIMAs, a seasonal extension of an ARIMA, remove trends in the data and post forecast errors to adjust for the moving averages. This approach utilizes differencing as the drive for trend analysis as opposed to exponential smoothing. However, SARIMA models do require stationarity – which is likely unrealistic given the volatility and non-linearity of energy consumption data.

Nonetheless, SARIMAs have shown considerable success – for example Nguyen et al, (Q. D., Nguyen, N. A., Tran, N. T., Solanki, V. K., González Crespo, R., & Nguyen, T. N. A. 2021) apply it to hourly electricity data from Northern Vietnam. Interestingly, they split the data into 24 different times series – one hour each – and were able to predict each individual hour with an average MAPE of 4.57%. However, their dataset did not include weekends, and it is probable that weekends have a slightly different electricity consumption structure than weekdays. The results may not be as strong if weekend data was included.

Like **Holt winters**, **SARIMA**s may struggle with multiple seasonality's. Bozkurt, Ö. Ö., Biricik, G., & Tayşi, Z. C. (2017) similarly employed a SARIMA model for hourly electricity market data in Turkey – utilizing the traditional SARIMA method, but augmenting them with parametric factors like weather, temperature, and even currency. However, they still found that an Artificial Neural Network model outperformed a SARIMA, but this was largely due to SARIMA inability to distinguish working days from holidays. Explicitly stating these variables using a SARIMAX model could help alleviate this issue. Nonetheless, the neural network MAPE of 1.80% and SARIMA's of 2.60% highlights the advantages of a neural net model.

Other time-series methods like the **TBATs** method by De Livera, Hyndmen and Synder (2011) have been used. By using Fourier series with time varying characteristics and being able to account for holiday effects and non-seasonal integers, they were able to show greater success than with more traditional methods.

Electricity data is complex, and traditional time series models struggle to account for the noise in the data. Using Queensland National Electricity data Al-Musaylh, M. S., Deo, R. C., Adamowski, J. F., & Li, Y. (2018) found machine learning methods were more effective than ARIMAs. Tarmanini, C., Sarma, N., Gezegin, C., & Ozgonenel, O. (2023) suggested that Artificial Neural Network (ANN) models outperformed ARIMAs, largely due to ARIMA's inability to cope with irregular patterns.

2.2 Machine Learning Models

However, despite an **ANN** capturing non-linear trends more effectively, it was still not able to capture peaks of consumption. Becirovic and Cosovic (2016) were able to produce MAPEs of around 1% using a Neural Net model. However, they do have a significantly smaller data set than we do – only hourly data, for 4 months in 2014-2015, across only two seasons (Summer and Winter). ANN models do have their critics, with Darbellay, G. A., & Slama, M. (2000) stipulating that the autocorrelations in the time series were primarily linear, negating the need for more complex, less explainable models.

Random Forests are another learning method employed. Random Forests are an ensemble learning method, able to incorporate features, and do not require stationarity like SARIMAs. However, they do not explicitly capture seasonality, so it would likely benefit from features specifying such. Hadri, S., et al (2019) used Random Forests as one method to forecast next day consumption for office buildings. They achieved a MASE of 0.68 - however, this was the worst performing model, with SARIMA achieving 0.59.

XGboost has become a very popular machine learning method, with considerable success in short-term electricity consumption. XGboost can articulate feature importance, making it more interpretable and understandable to stakeholders compared to other machine learning methods. Using hourly records from Panama from

2015-2020, XGboost produced the lowest MAPE of 3.74% compared to other machine learning methods (Madrid, E., & Antonio, N. 2021). They particularly noted the impact of lag loads on future predictions, as well as having a holiday indicator. Zhao et al (2022) use an XG boost method on 30-minute windows, supplemented with humidity and temperature data, as well as implementing real-time electricity prices. Their original XGboost without using their windowed method produced MAPES of ~3%.

2.3 Impact of Feature Engineering

Beyond the models used, there is significant research on the variables that impact electricity consumption. Attention has been given to calendar effects. For example, Arora and Taylor (2018) propose a method that accommodates 'special day', such as holidays, by using indicator variables to account for their changes in demand. For our NSW data, this highlights the need to look at holidays and their different consumption profiles. Additionally, Fahad and Arbab (2014) look at a time factor – I.E. day of day, week of day, etc. They propose that time could be categorized by 'Working Time', 'Leisure Time' and 'Sleeping Time' as indicator variables, and used to help predict consumption. Equally, this draws attention to potential 'lagged' variables that may be useful as analysis, such as the serial correlation electricity consumption and hours in the day.

In terms of exogenous variables, intuitively it seems that temperature would have the greatest impact on electricity use. While Nti, Teimeh, Nyarko-Boeteng & Adekoya (2020) state in their systematic review of electric load forecasting that temperature at times has no impact, they stipulate that it may be related to economic conditions. While not elaborated, it stands to reason that in Australia that more extreme temperatures would lead to different electricity consumption – in hotter times, more aircon, and colder times more heating. Several studies have investigated weather impacts – with Wang, Zhang and Chen (2021) studying how weather impacts the per minute electricity load for 114 single-family apartments in China. Their results indicated that, overall, weather did impact electricity consumption but noted that it was not a universal effect. Abu-Salih, et al (2022) collected real time energy consumption from residential premises in Western Australia. While they do not explicitly state the significance of temperature on their analysis, they present a correlation graph with temperature and electricity consumption, which shows an overt trend. Hence, temperature is likely to be an important factor in our analysis.

3. Data and Methodology

3.1 Software and Tools

We will be using python and Jupyter notebook as the primary tools for our analysis. The flexibility of python to perform different functions along with our overall familiarity with it has made it the clear choice. Jupyter enables us to perform extensive data manipulation and modeling in a retainable and reproducible manner.

We will be using several python packages including; Numpy, Pandas, Seaborn, sklearn, and tensorflow. These will facilitate data handling, visualization, and model development.

NumPy is employed for efficient numerical computation, while Pandas provides structures for data cleaning, manipulation, and analysis. These packages also serve as the basis for the feature engineering our models depend on.

3.2 Data Description

Our data consists of Temperature, Demand, and Forecast Demand broken down by location and time. The location for all data points is Bankstown and we will be using it as a proxy for greater New South Wales. Our data is in approximately 200000 half hour increments from 01/01/2010 to 31/03/2021.

3.3 Data Cleaning

The first step in any model development is cleaning the data. This ensures that data is consistently formatted and free of any technical details that could cause errors.

The first step in our cleaning is to remove whitespace from the datetime column and standardize it to the correct format. We have chosen to use *"dmy hm"* format. In this process forecast demand is aggregated by the mean. We identified 13 duplicate values in our data set, we kept the first of these and dropped the remaining 12.

We also identified a number of missing 30-minute intervals. After these were visually inspected, we filled them using the built-in pandas *'ffil'* function. This function takes the previous valid observation and applies it to the erroneous observations.

The data for the year 2021 is only present to the end of march, and as such we have decided to exclude these 3 months of data. This ensures that our data set only includes complete years.

3.4 Feature Engineering

We determined that there would be a number of variables we could produce from existing data that would allow us to produce a better model.

We then created a number of time-based variables; *Month* to track which month a data point is from, *Week of year* to track how many weeks into the year a data point is, *Day of Week* to track the day of the week, and *Hour* to track which hour of the day a data point is from.

We created Boolean flags for summer and winter. The intention of this was to track what calendar season data is from, we believed that there is likely a substantial seasonal difference for the trend in electricity demand. We have defined summer as November to March and winter as June to August. Months outside of these are not flagged as part of either season.

We tracked what the peak hour of each day is divided by season, i.e. what the peak hour is for Summer or for Winter. From here we created a Boolean to measure whether a datapoint occurs during the peak hour.

We created two Boolean variables, *Weekend* and *Public Holiday* to track whether data is from a weekend or a public holiday. We theorized that usage would be different on these days as the places people spend time, home vs work or school, would be different.

We also created multiple auxiliary numeric variables; Lag for both 24- and 48-hour periods. These variables store respectively the demand for the previous 24 and 48 hours, Sin and Cos transformations for both day and year to better capture cyclical patterns in time, finally we created a square of temperature to better encapsulate potential non-linearity in the relationship between temperature and demand. A complete summary of the feature details is provided in **Appendix A**.

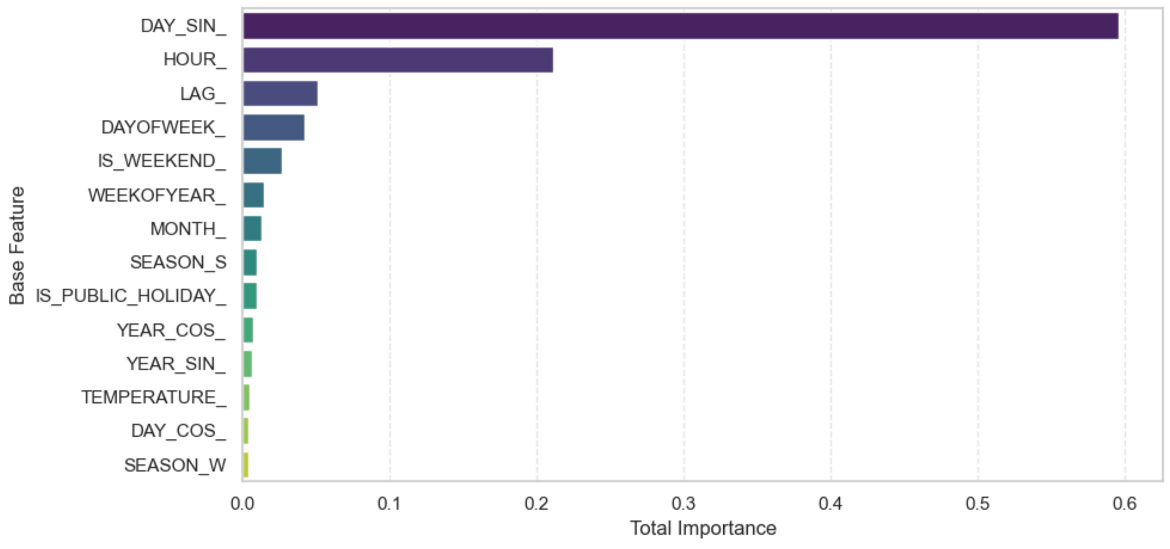


Figure 1. Feature Importances. A complete feature importance summary is provided in ***Appendix B****.*

3.5 Sliding Window and Forecasting Horizon

In time series analysis, the sliding window technique allows models to observe past data of a pre-determined size to make a future prediction. This process includes splitting data into chunks, which helps reduce computational complexity by focusing on smaller amounts of data. **Forecasting Horizon** is predicting (for instance) one hour ahead

e.g. in our case, a window size of 24 means that the model uses the past 24 hours of electricity demand to make a prediction. The model is still trained on the entire dataset, but each training sample is created using 24 input hours to predict. That process continues over the entire dataset and the model learns patterns across thousands of windows.

Table 2. Illustration of the Sliding Window Approach for 1-Hour Ahead Forecasting. Each training sample uses the past 24 hours of electricity demand as input to predict the demand 1 hour ahead. As the window slides forward one step at a time, new input-target pairs are generated.

Sample #	Input (24 hours)	Target 1 hour ahead
1	Demand from t = 1 to t = 24	Predict at t = 27
2	Demand from t = 2 to t = 25	Predict at t = 28
3	Demand from t = 3 to t = 26	Predict at t = 29

The figure below illustrates how a window of size 4 and a forecasting horizon of 3 operates shifted forward by one time step at each iteration.

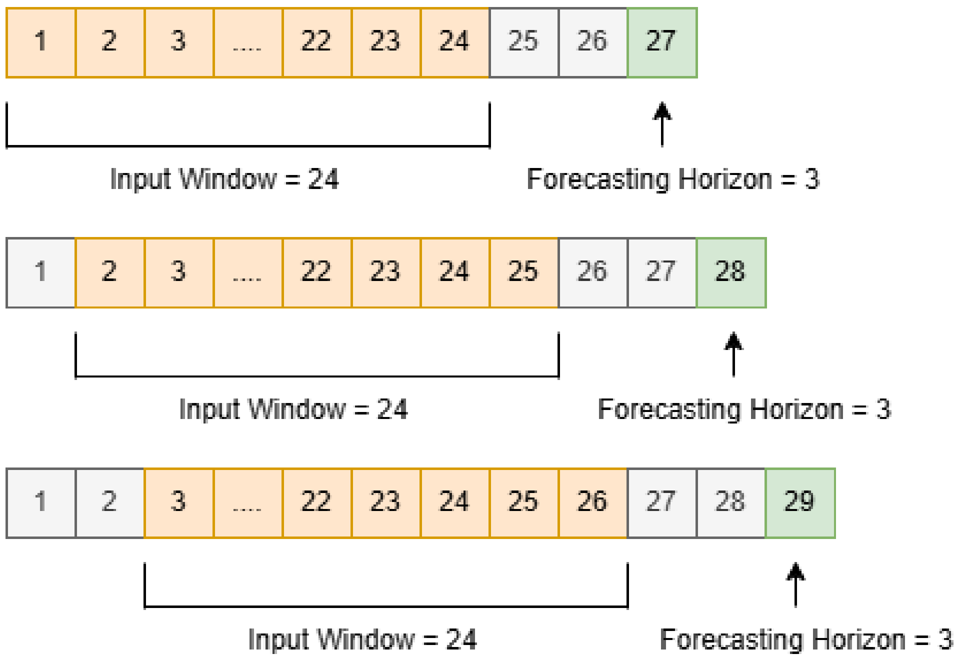


Figure 2: Sliding Window and Forecasting Horizon

3.6 Modeling Methods

We will apply several modelling methods to the data in an attempt to generate the most accurate predictor that we can within the available constraints.

Holt-Winters Model:
The Holt-Winters model is an extension of exponential smoothing. Exponentially smoothed models assign greater weight to more recent measurements than older ones. Holt winters extends this model so that it can account for seasonality.

Holt winters uses three smoothing parameters. Alpha, the level smoothing parameter, controls the speed at which the model updates. Higher values place more weight on recent observations and vice versa. Beta, the trend smoothing parameter, controls the speed at which the model updates its trend estimation. Gamma, the seasonality parameter, refers to how sensitive the model is to seasonal patterns.

SARIMA Model:
SARIMA (Seasonal Autoregressive Integrated Moving Average) is an extension of the ARIMA model that allows for the existence of seasonality in data sets. The SARIMA model has 6 components: p, d, q, P, D, Q. These correspond as follows; p is the non-seasonal autoregressive term representing the number of lagged observations to use, D is the order of integration representing the number of differences between the current value and past values required to make the model stationary, q representing the dependency between the current value and the residual error from past predictions. P, D, and Q are the seasonal equivalents of the previously described components.

Long Short-Term Memory networks (LSTMs):
Recurrent Neural Networks (RNNs) are a type of neural network designed to handle sequential or time series data. They can analyze time series data efficiently; therefore, they are widely used in projects such as, hourly temperature forecasting, speck recognition and daily energy consumption. Unlike traditional neural networks, RNNs are able to predict the future by linking past information with the present. However, standard RNNs often struggle to recognize differences in which information is important, especially when dealing with long-term dependencies.

Long Short-Term Memory networks (LSTMs) are a type of RNN designed to address the standard RNNs limitations by avoiding the long-term dependency problem. LSTMs can learn and understand long-term patterns. Both standard RNNs and LSTMs use a repeating module that helps maintain a connection with past inputs.

3.7 Modeling Methods

To evaluate the quality of our predictors we will use a number of numeric evaluators such as mean square error. These evaluate predictions based on how distant they are from the true value. For example, mean square error takes the mean of the square of error values. This punishes values more the further they are from the true value. Another important type is the MAPE, *mean absolute percentage error*. This evaluates the mean percentage distance a prediction is away from the true value. This avoids downplaying errors in smaller values where absolute errors can appear smaller but in reality, be as large of a forecasting error. We will also use learning curves as a visual evaluation method. These will inform us primarily as to whether our model is under or overfit to the training data.

4. Exploratory Data Analysis

4.1 Data Characteristics

We see a relatively stable amount of electricity use between 0 and 20 degrees. Beyond this point usage starts to steadily climb. There is some seemingly random fluctuation above 40 degrees, this is likely due to a lack of data.

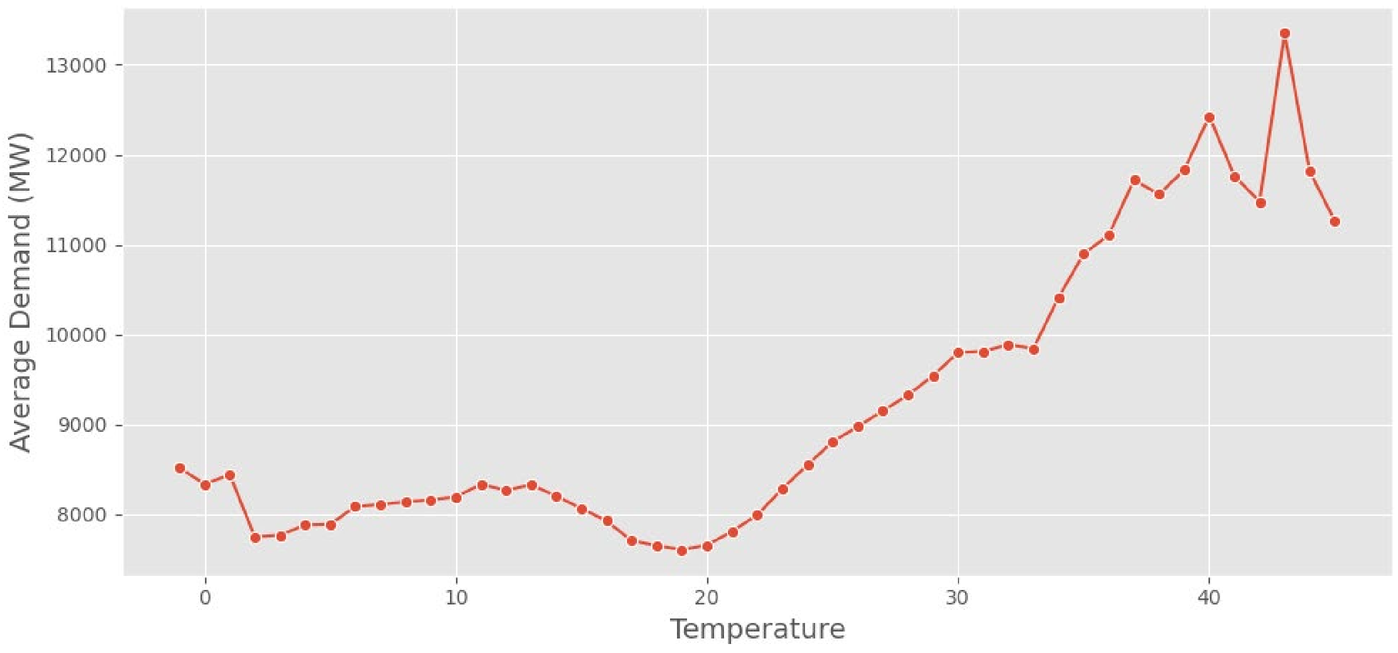


Figure 2: Demand by Temperature

Dividing this further by season we see no unexpected trends.

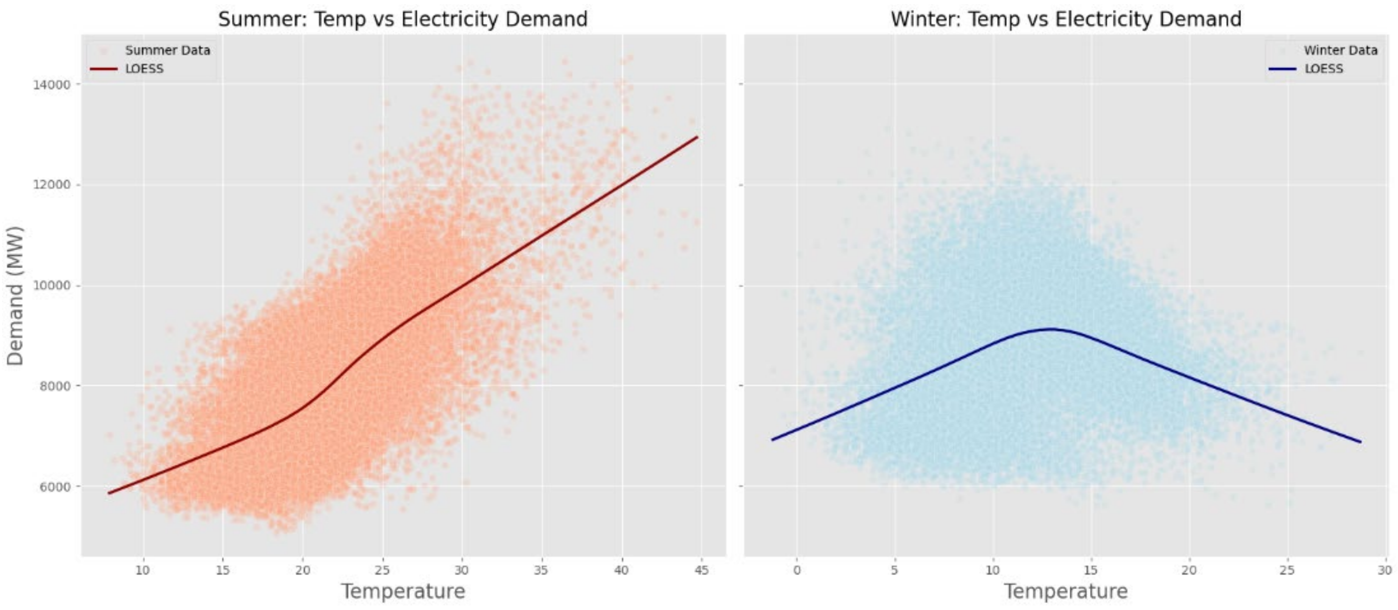


Figure 3: Seasonal Demand by Temperature

We see no meaningful differences in yearly trends over the range of our data set.

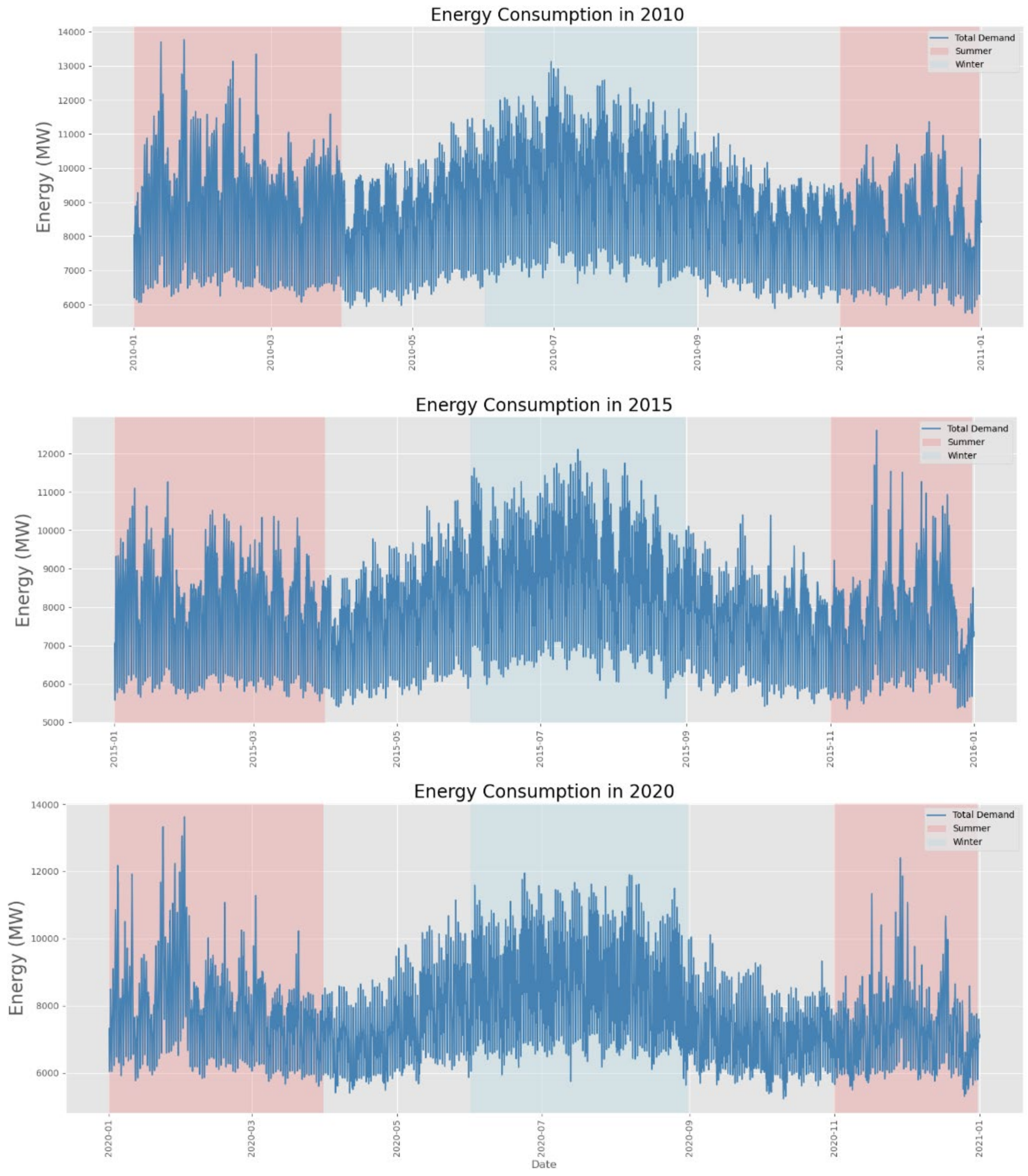


Figure 4: Consumption by year

4.2 Forecast error analysis by day type and season

We see the greatest forecasting errors in summer. We also see dramatically higher forecast errors for public holidays and school breaks than for regular days.

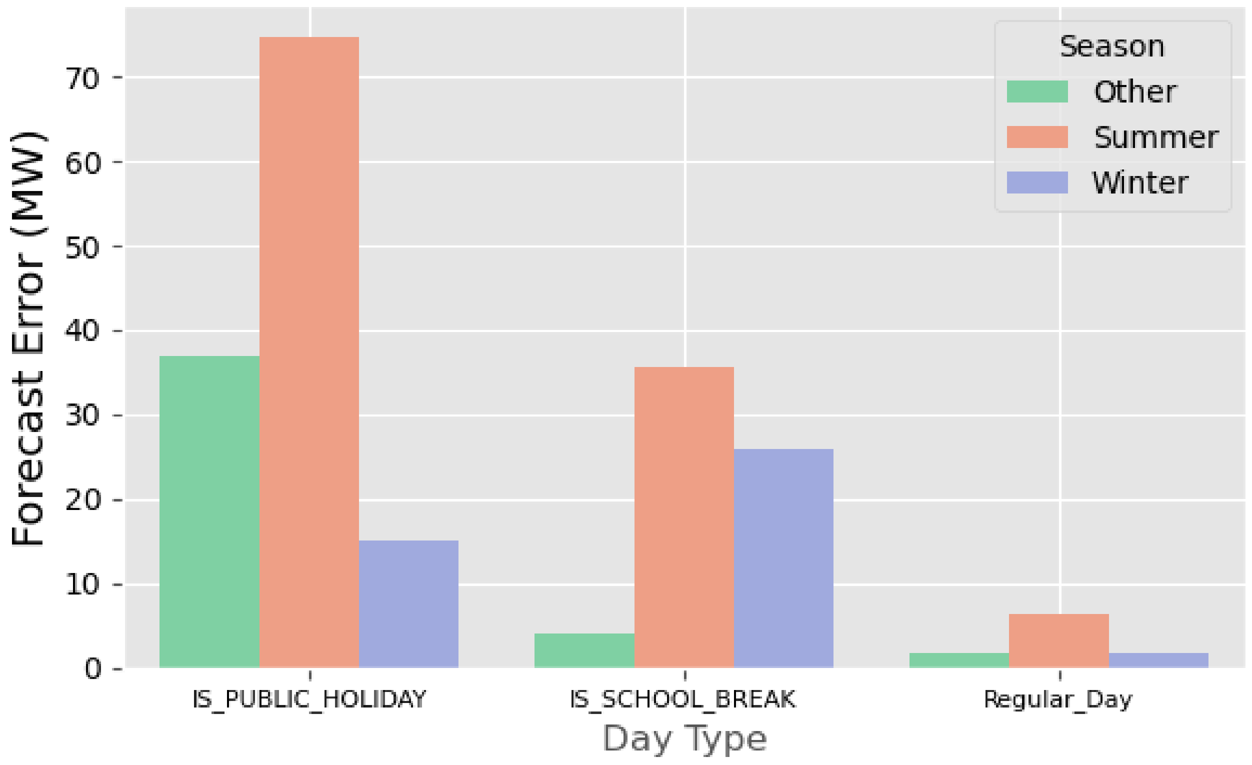


Figure 5: Forecasting Error by Day Type and Season

We also see that the largest forecasting errors happen during summer.

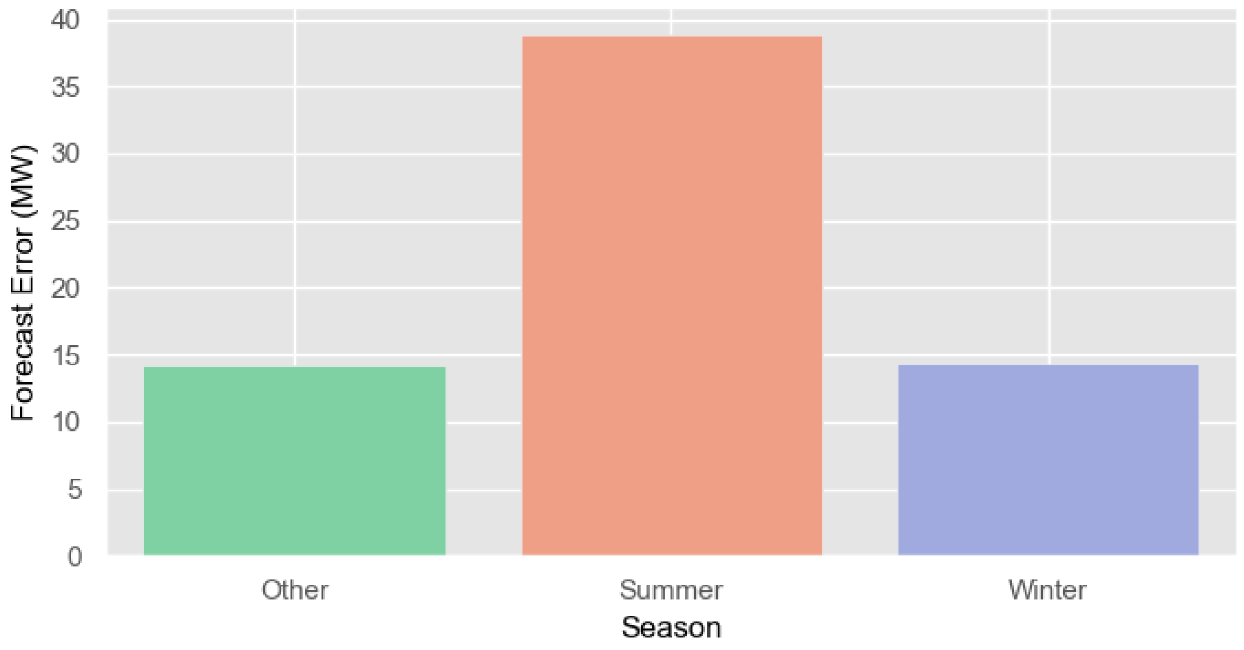


Figure 6: Forecasting error by Season

There is a noticeable difference in the pattern of usage over a day between summer and winter. In summer usage is relatively consistent between 8 am and 7 pm with a slight dip outside of those hours. In winter there is a substantial dip after an 8am peak before another peak between 6 and 7 pm.

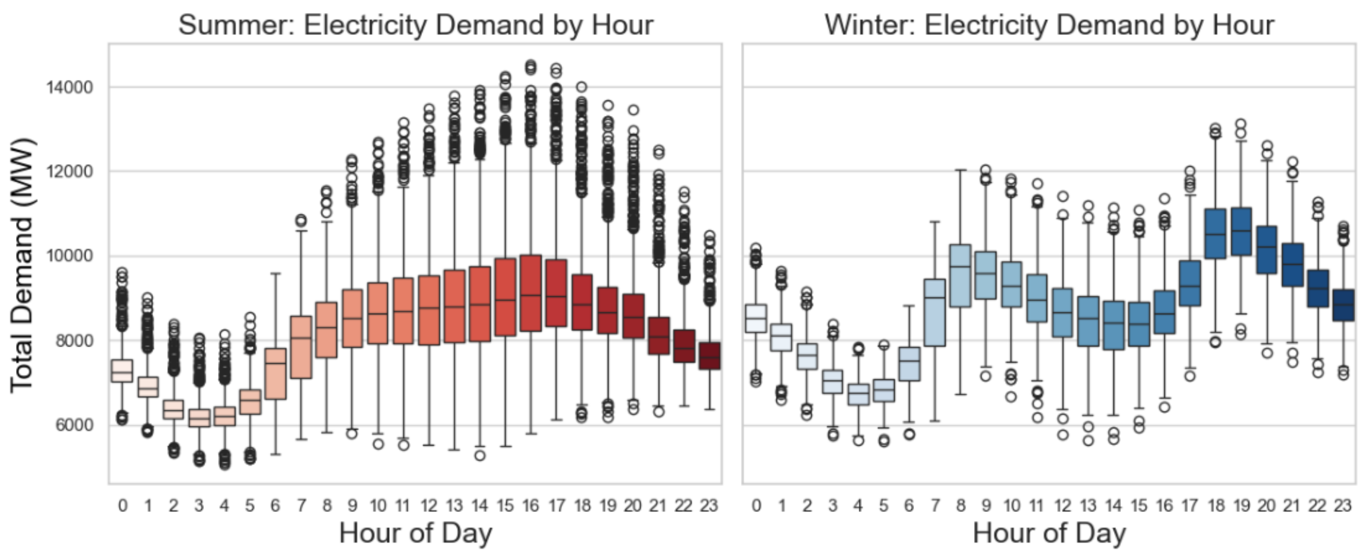


Figure 7: Seasonal Demand by Hour

For additional plots illustrating the relationship between Total Demand and Temperature, Hour, Week and Month, refer to Appendix C

5. Model Development and Evaluation

We decided initially to attempt to use statistical models to generate our predictions. Our thought was that the interpretability of these would allow us a greater understanding of what drives demand alongside producing numerically accurate predictions. This interpretability would be valuable to stakeholders' decision-making processes.

We selected Holt-Winters and Sarima as the statistical models to experiment with. We also expected a neural network to produce better numerical outcomes at the cost of that interpretability and as such worked to develop one in parallel.

5.1 Holt Winters

Holt winters models are a suitable choice for this data set for its ability to handle trends and seasonality. Unfortunately, the nature of Holt Winters prevents us from including potential explanatory variables such as temperature, instead relying solely on trends over time for prediction.

To begin we split hourly data into 80%/20% training and test sets. The model gave coefficients of:

- Alpha (level): 0.8182
- Beta (trend): 0.0001
- Gamma (seasonality): 0.1818

We have a model that adapts very quickly to changes at the base level making it responsive to short term variation, however, the low beta and gamma indicates that the model is not updating the trend heavily year to year and season to season.

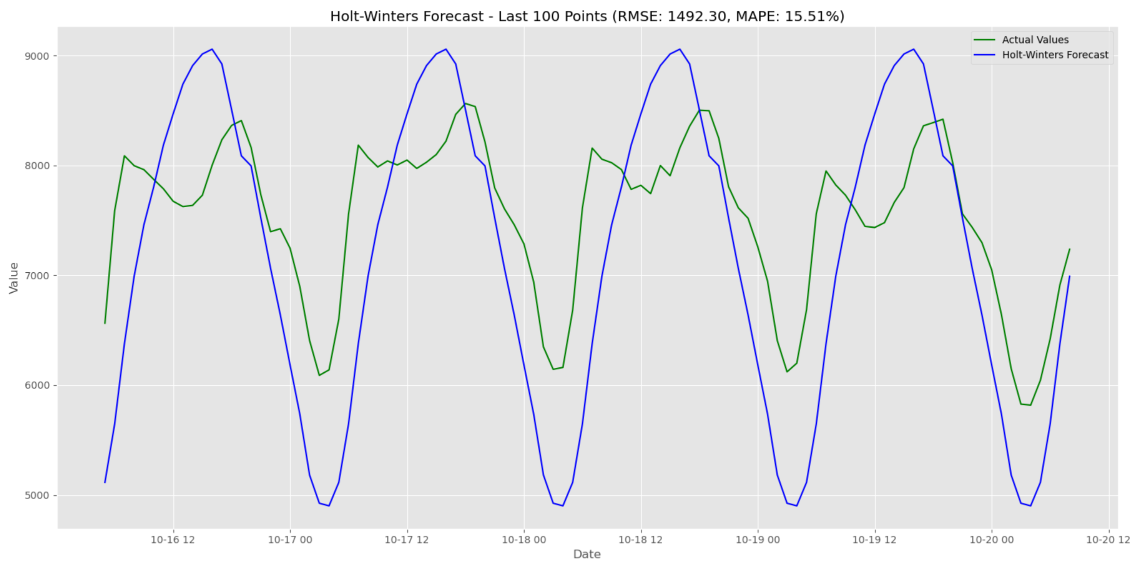


Figure 8. Holt-Winters over 100 data points

We observe that the Holt Winters model approximates the shape of the data quite well. The model also visibly fails to account for small scale variation in the data. Over this time period the MAPE is 15.51% which is quite poor.

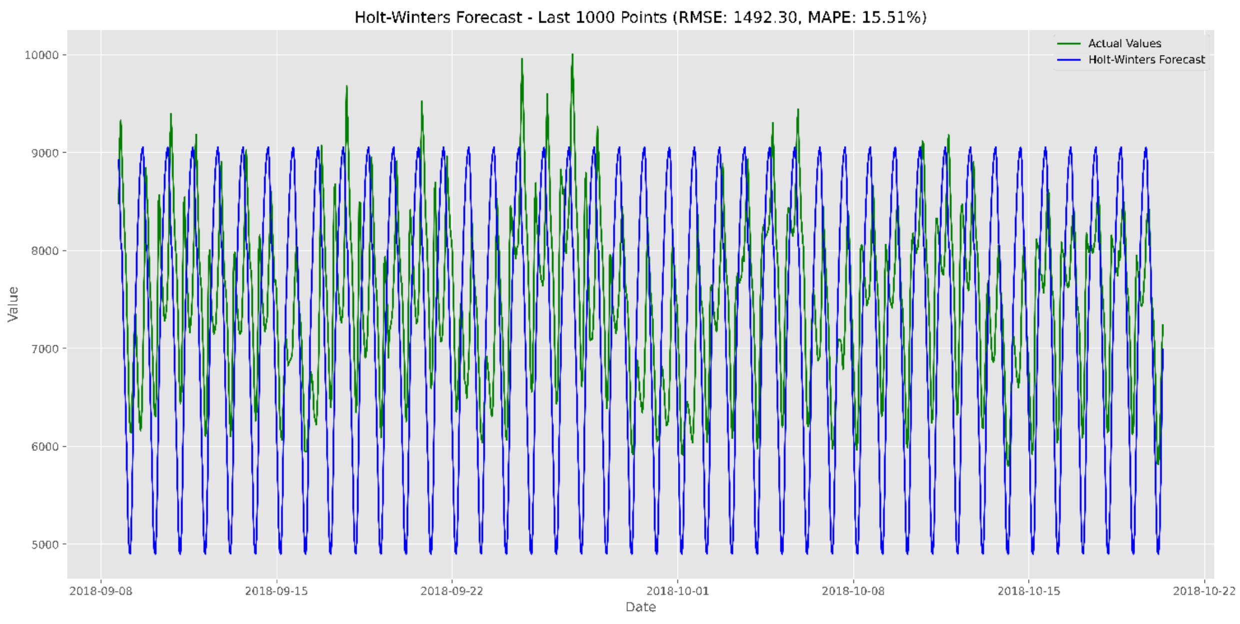


Figure 9. Holt Winters over 1000 data points

Over 1000 data points we see the same visible flaws and have a MAPE of 15.51% once again. The primary flaw in the holt winters is the overly uniform nature of the predictions.

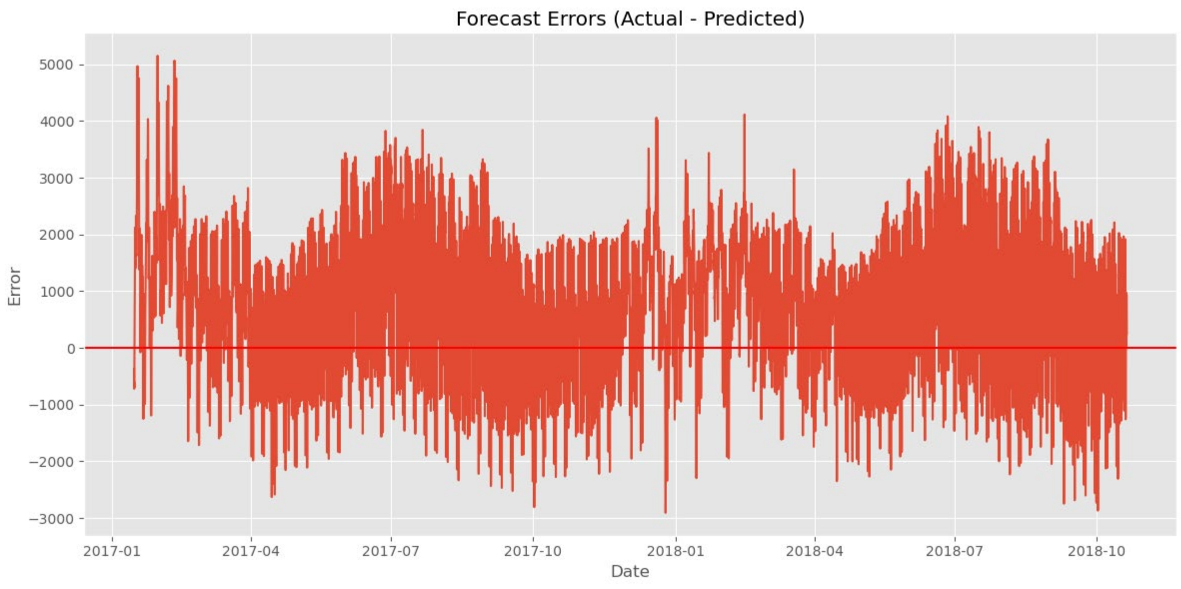


Figure 10. Holt winters error plot

The error plot demonstrates the model is both over and underestimating values. In combination with the high MAPE we have enough information to conclude that Holt Winters is not a sufficiently accurate method for generating predictions in our use case.

5.2 SARIMA

Sarima Models are a popular choice for application to time series data as they explicitly account for seasonal components and incorporate exogenous variables to allow for more complex data.

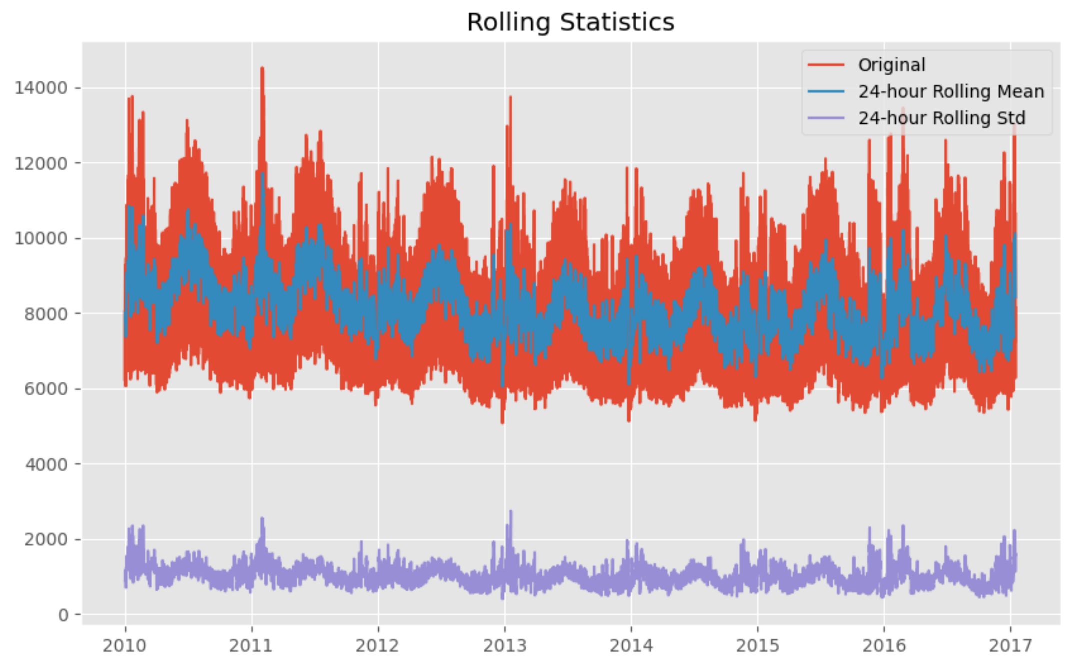


Figure 11. Rolling Statistics

Sarima models function best when the underlying data is stationary. Visual inspection indicates that the data may be stationary but a kpss test gives a score of 16.1874. KPSS Tests check the null hypothesis that a series is stationary against the alternative hypothesis that it is not, a score of 16.1874 tells us our data is not stationary.

To deal with the non-stationary data we perform a first differencing. First differencing, our data converts it from the raw numbers to the series of changes from one point in time to the next.

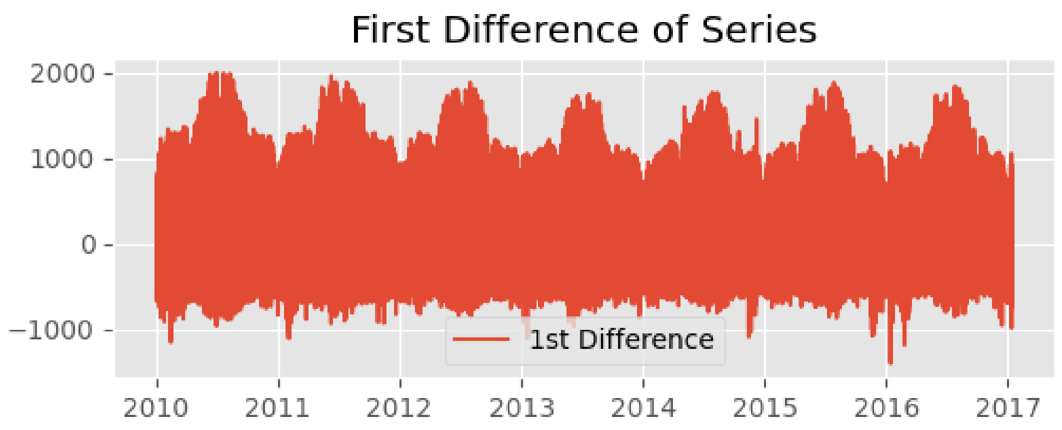


Figure 12 First Difference of Series

After first differencing our data is suitable for use in a SARIMA. There are two numeric components that must be chosen for a SARIMA: P the non-seasonal autoregressive component, and Q the non-seasonal moving average component. To select these, we examine the auto and partial correlation plots.

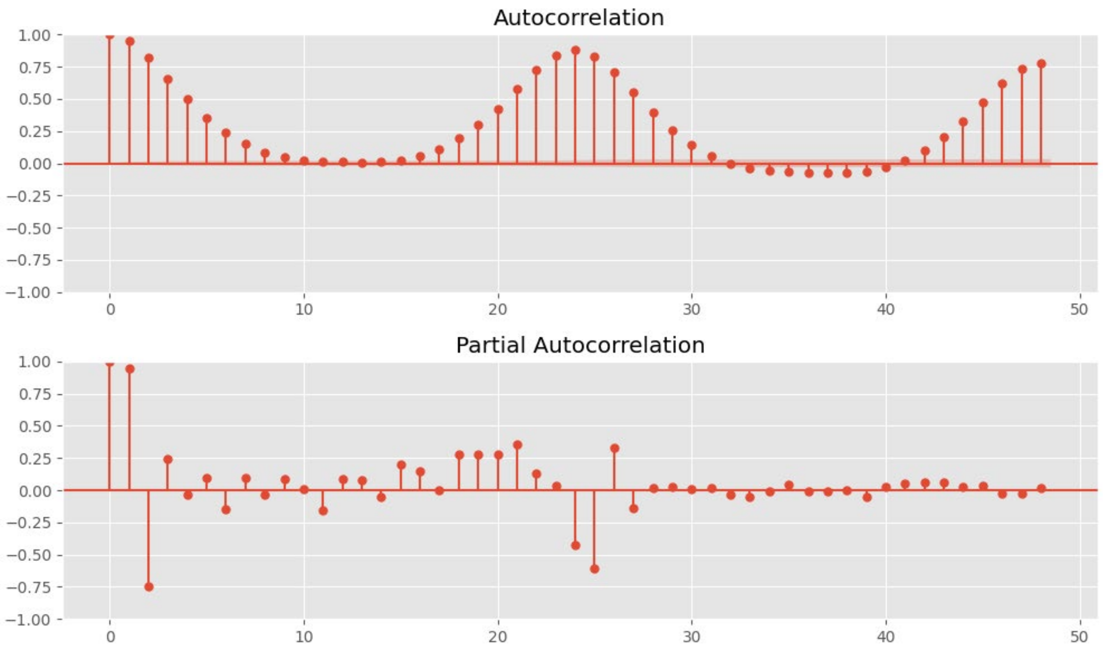


Figure 13. Correlation Plots

There is a clear daily autoregressive component with peaks through 24-hour intervals. Differencing the data by an additional 24 hours should mitigate this.

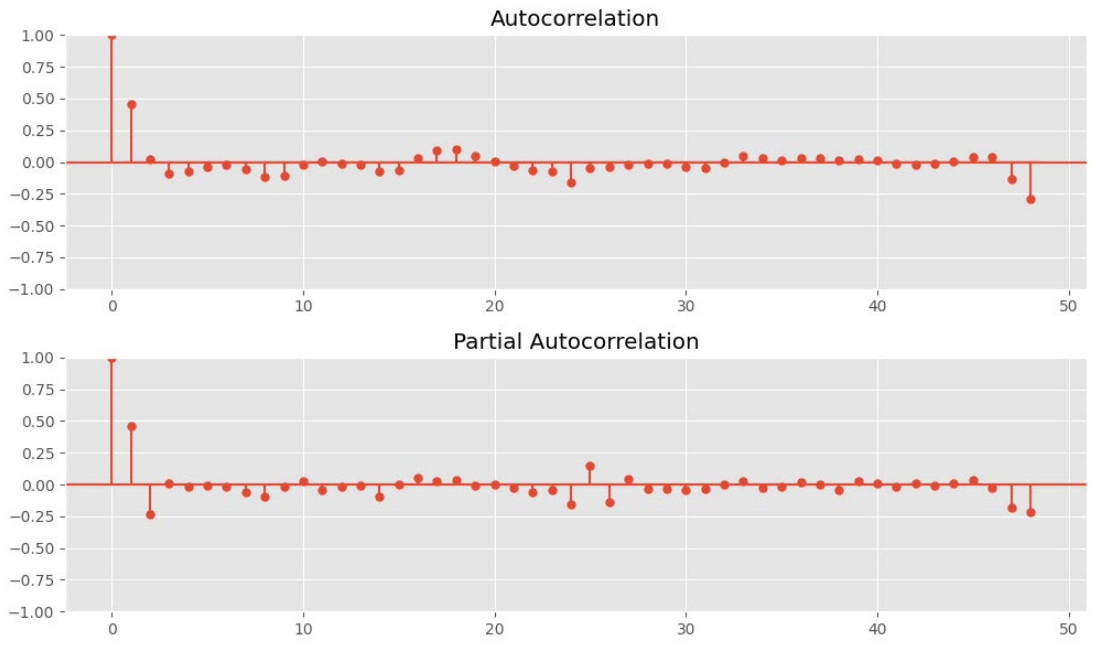


Figure 14. Correlation plot after 24-hour differencing

We now observe both ACF and PACF cut-offs at approximately 2, as such we will set both P and Q to be 2.

We now move on to accounting for potential seasonality. Given our previous observations we will begin with a seasonal component of 24, we also set the D parameter to 1 to remove that seasonality. We also confirm that these parameters work best through a grid search.

Applying this to our model we achieved a MAPE (Mean Absolute Percentage error) of 3.5% on the validation set.

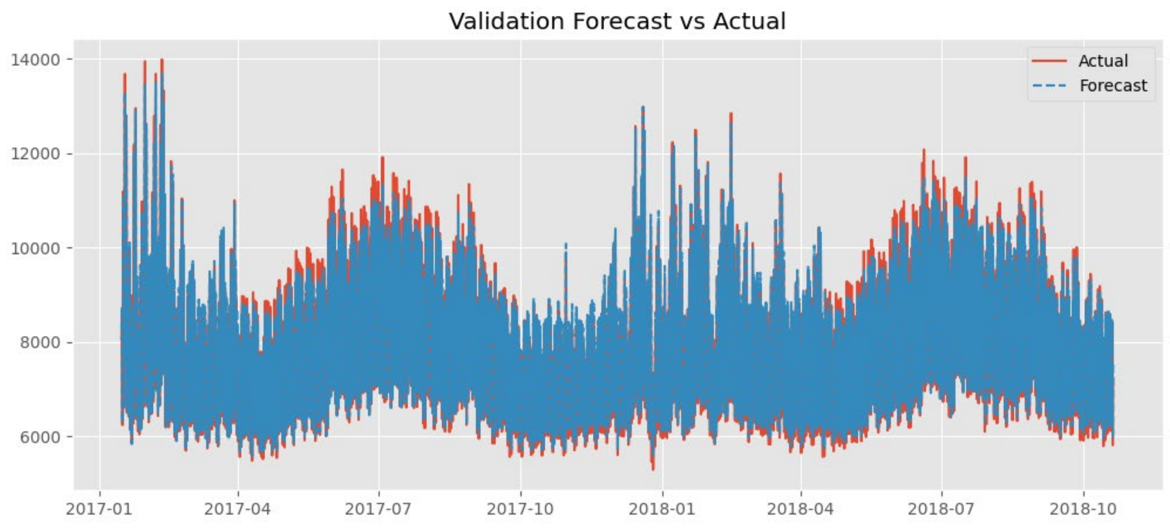


Figure 15 Forecast vs validation data

The SARIMA estimates were very sensitive to changes in the amount of training data used. This is likely due to the covariance matrix being singular or near singular. While we could resolve this issue with PCAs and regularization, the more complex neural network we develop will be more effective at handling these potential issues.

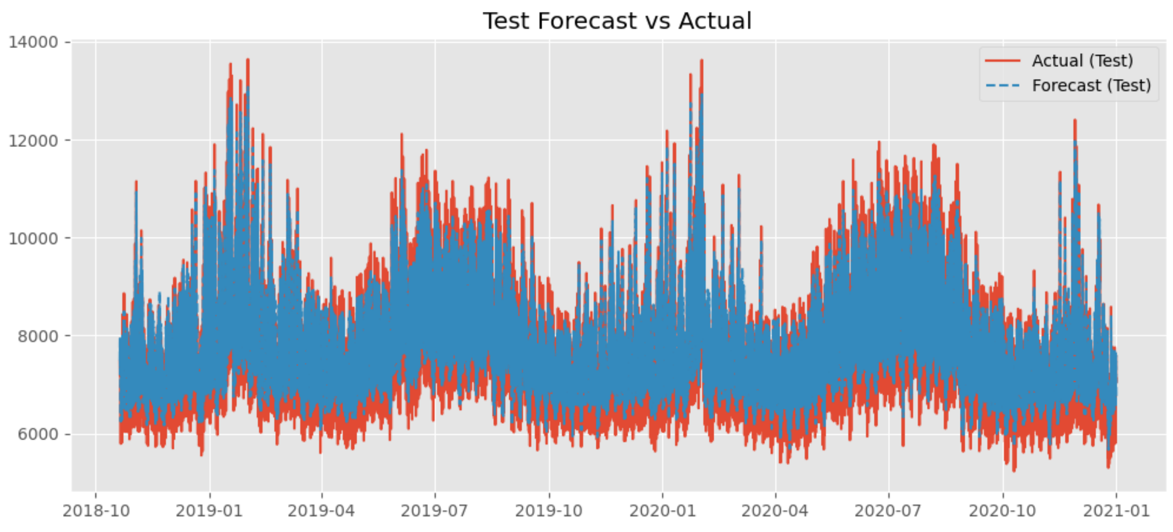


Figure 16. Forecast on test data

We observe a strong fit of our model; however, we also note that the model fails to address the largest peaks and troughs in the data. The model produces a MAPE on 5.2%

on the test set. Our exogenous variables: Temperature, weekend, lagged demand are all significant, validating the feature engineering we performed.

We also observe randomly distributed residuals suggesting a model that more robust modeling is likely to produce better results.

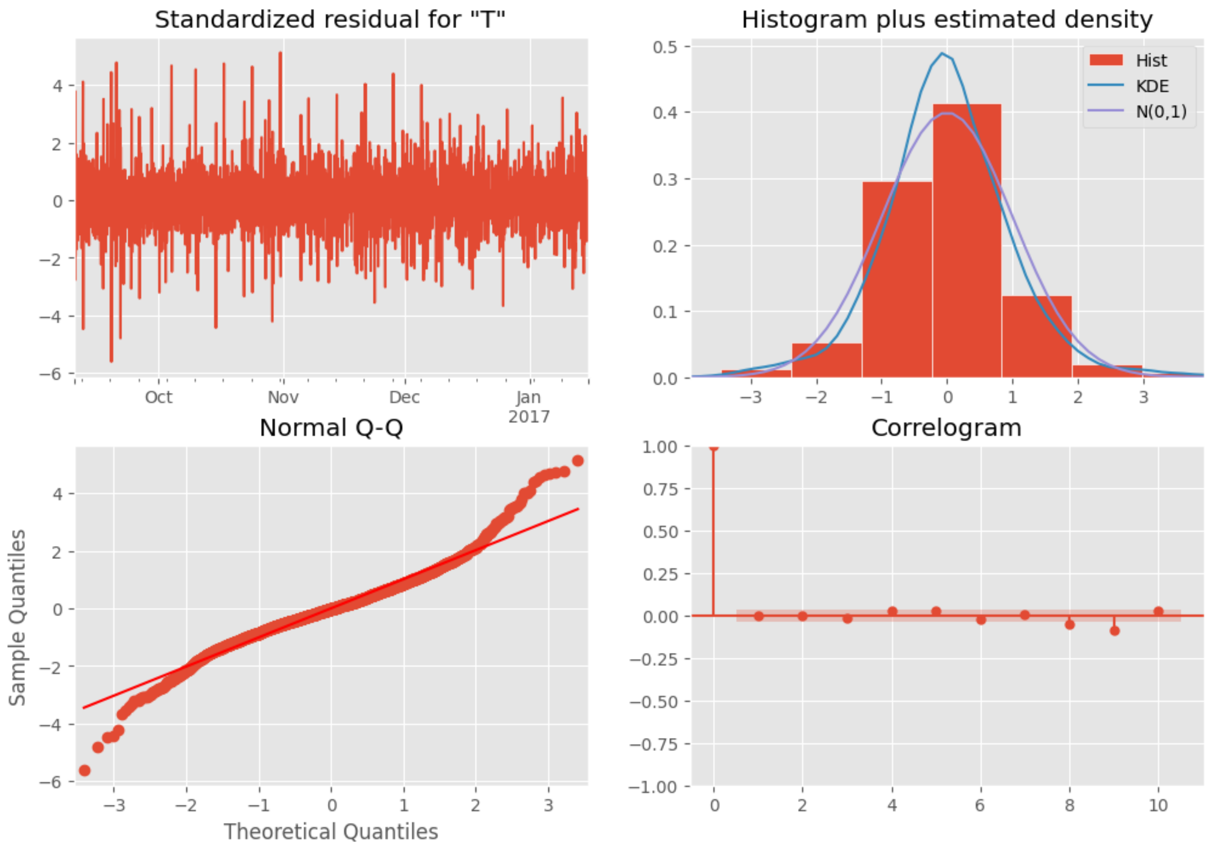


Figure 17. Residuals

5.3 LSTM (Long-Short-Term Memory)

Our development of statistical models has consistently suggested that a more comprehensive model could be developed. We have selected an LTSM neural network as the method most likely to give us strong results.

As the focus is short-term forecasting, we decided that a 1-hour horizon would most appropriately allow us to meet stakeholder needs. We used input windows of 24, 48 and 72 hours. What this means in practice is that we predict one hour based on the determined lookback duration (24, 48 and 72 hours).

We begin with a basic TensorFlow sequential model. This model has a plain stack of layers where each layer has one input and output tensor (TensorFlow, n.d.) illustrated in Figure 18, The model architecture, consists of a single LSTM layer with (128 units or 64 units), followed by second LSTM stacked layers (with 64 units) to reduce complexity and allow the model to capture more complex patterns.

A Dense layer was added after the second LSTM to enhance the model's ability to learn complex feature interactions. This helped the model better understand non-linear relationships between features, ultimately improving prediction accuracy.

We also used a Dropout layer, which is essential in any LSTM-based model. Dropouts help improve generalization, which helps reduce the risk of overfitting.

For a detailed description of the LSTM architecture and mathematical formulation, refer to **Appendix E**.

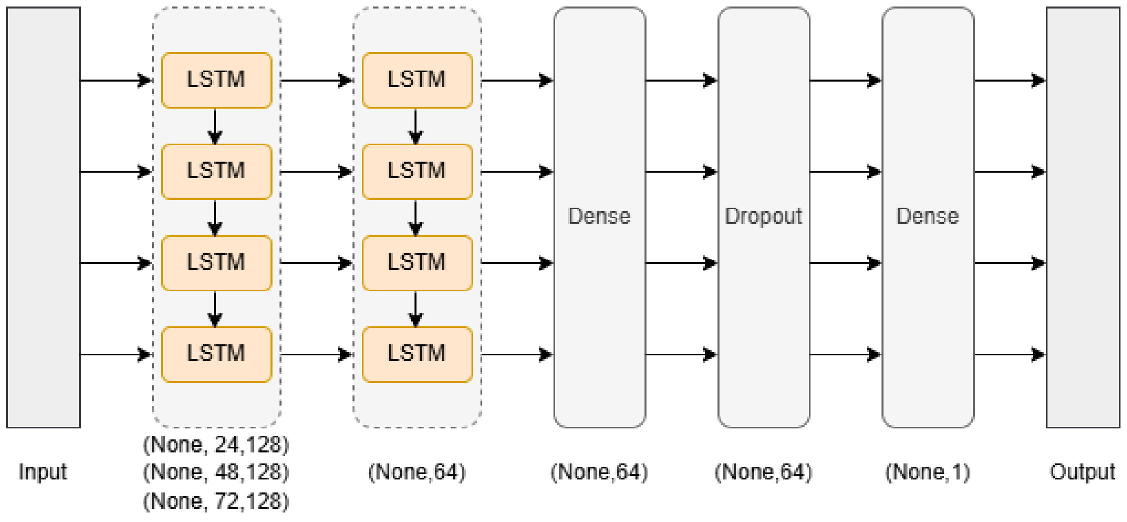


Figure 18. LSTM model architecture

The model was trained using the Adam optimizer over 100 iterations, with early stopping enabled. We also used **Optuna** -- which is an open-source framework for automated hyperparameter optimization -- to find the best combination from the list below:

Table 2. Hyperparameter search space setup for LSTM model optimization. This table lists the candidate values explored during the hyperparameter tuning process using Optuna.

Number of units	[32, 64, 128]
Dropout Rate	[0, 0.2, 0.5]
Learning Rate	[1e-4, 1e-2]
Batch Size	[32, 64, 128]

The final model was trained with the best combination of parameters found using Optuna. It used a window size of 24, 48 and 72 hours, and a total of 100 epochs,

LSTM Forecast Performance by Window Size

We conducted a manual search to identify the best window size between 24, 48 and 72 hours for the 1-hour forecast horizon. Model performance was also evaluated using architectures with 2 stacked LSTM layers, while keeping the learning rate fixed at 0.001 as determined through the Optuna tuning process. This evaluation included a variety of dropout rates, node numbers and batch sizes, which were identified in the previous step.

Best Model Results

Table 6. The table presents the LSTM forecast best performance with **Dropout = 0**. It summarizes the best evaluation metrics (MAPE, MAE, RMSE, and R²) results across 24, 48 and 72 window size

Hyperparameters				**LSTM Results**			
Window Size	**Nodes #**	**Dropout**	**Batch size**	**MAPE (%)**	**MAE**	**RMSE**	**R²**
24 Hour	128	0	32	3.45	263.35	367.93	0.91
48 Hour	128	0	32	3.23	246.82	346.83	0.92
72 Hour	128	0	32	3.22	246.53	344.01	0.92

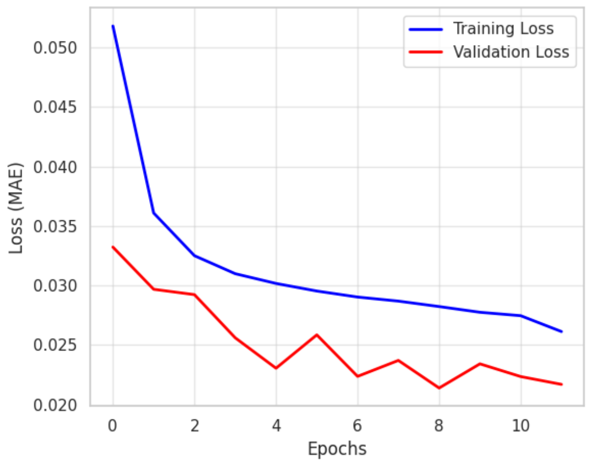
Table 7. The table presents the LSTM forecast best performance with **Dropout = 0.2**. It summarizes the best evaluation metrics (MAPE, MAE, RMSE, and R²) results across 24, 48 and 72 window size

Hyperparameters				**LSTM Results**			
Window Size	**Nodes #**	**Dropout**	**Batch size**	**MAPE (%)**	**MAE**	**RMSE**	**R²**
24 Hour	128	0.2	32	4.14	314.39	444.37	0.87
48 Hour	128	0.2	64	3.76	287.03	411.38	0.89
72 Hour	128	0.2	32	3.73	285.75	410.17	0.89

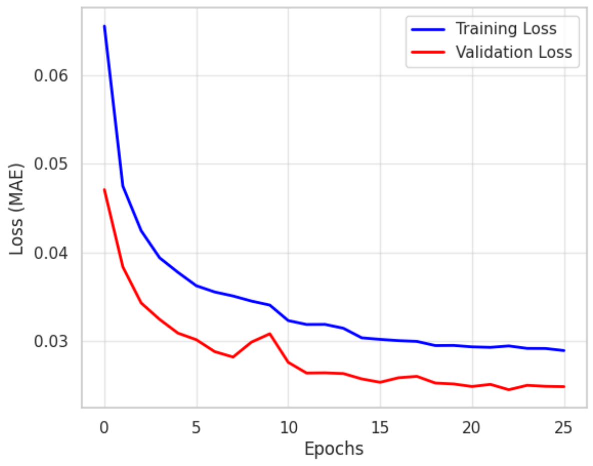
Regarding hyperparameters, models with 128 nodes consistently outperformed those with 64 nodes, achieving lower MAPE and MAE values and higher R² scores across different window sizes. This suggests that using a larger number of neurons improved the model's ability to capture complex patterns. In terms of regularization (Dropout rate), models trained with dropout = 0 achieved better results than those with a 0.2 dropout rate.

The 24-hour window and dropout rate of 0 achieved good results, with R² around 0.91. However, its MAPE and MAE values were slightly higher compared to longer look-back periods, such as 48 and 72 hours, indicating that 24-hour look-back may not capture longer-term patterns efficiently. In contrast, both the 48-hour and 72-hour windows delivered better performance than the 24-hour, achieving lower MAPE and MAE and higher R². Both the 24-hour and 48-hour windows achieved similar accuracy, with less than 1% difference across all evaluation metrics. The 48-hour lookback offers a lower computational cost while still enabling the model to learn complex patterns effectively. LSTM training results and tested hyperparameters for different window sizes are summarized in **Appendix F**.

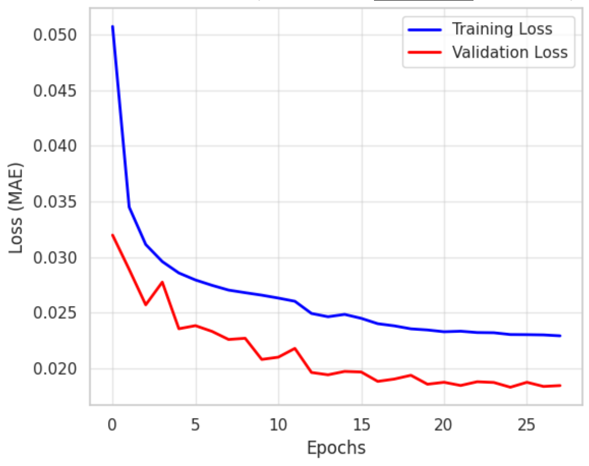
Window size: **24-hour** (128 nodes, **<u>0 dropout</u>**, batch 32)



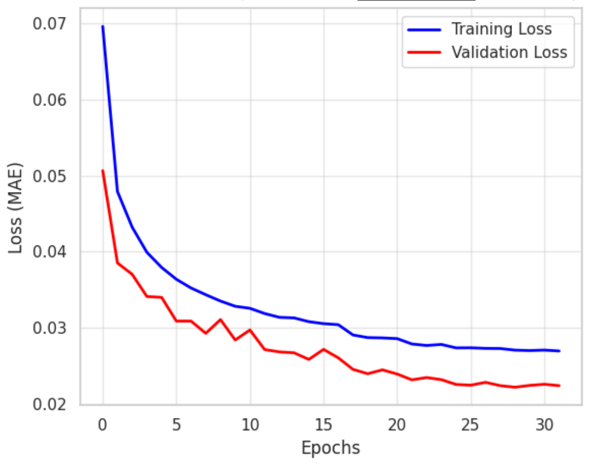
Window size: **24-hour** (128 nodes, **<u>0.2 dropout</u>**, batch 32)



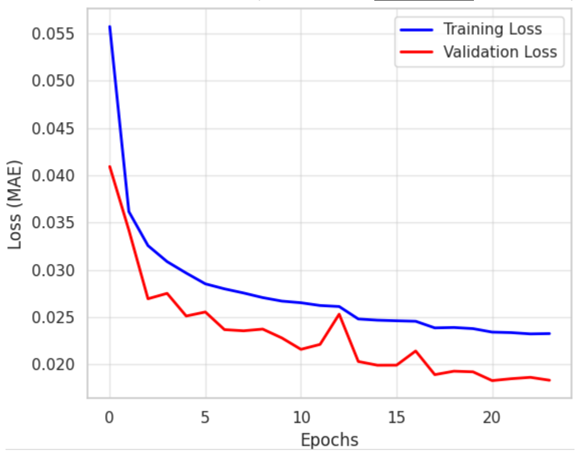
Window size: **48-hour** (128 nodes, **<u>0 dropout</u>**, batch 32)



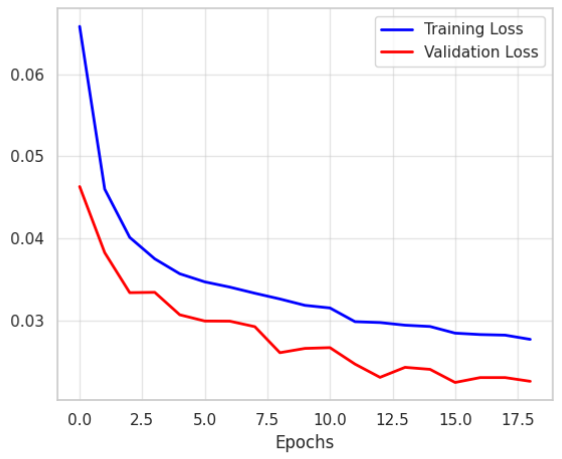
Window size: **48-hour** (128 nodes, **<u>0.2 dropout</u>**, batch 64)



Window size: **72-hour** (128 nodes, **<u>0 dropout</u>**, batch 32)



Window size: **72-hour**(128 nodes, **<u>0.2 dropout</u>**, batch 32)



Learning curves are essential tools for diagnosing overfitting and underfitting in machine learning models (Ibrahim, 2023). In this analysis, we used learning curves to evaluate model behavior across different window sizes (24, 48, and 72 hours). The gap between the training and validation losses was not large, indicating no major issues with generalization, as the models were able to generalize their learnings from the training data. However, both the 48-hour and 72-hour models achieved final validation losses around 0.022, demonstrating stronger forecasting performance and generalization compared to the 24-hour window, which finished with a higher validation loss between 0.025 to 0.028.

While the 72-hour model achieved slightly lower forecasting errors, the 48-hour model showed smoother convergence behavior with fewer fluctuations in the learning curves. The distance between the training and validation losses appeared smaller for the 48-hour model with a 0.2 dropout rate, while for the 72-hour model with 0 dropout, the gap also looked small.

In general, the plots did not show signs of overfitting or underfitting, as both the 48-hour and 72-hour models performed well. However, the 48-hour learning curve appeared more stable compared to the others.

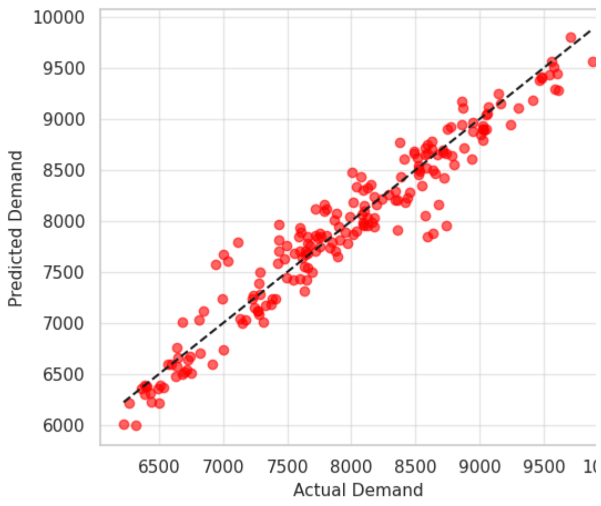
The scatter plots below show that the 24-hour, 48-hour and 72-hour forecasts demonstrate strong alignment between predicted and actual values, with tight clustering along the diagonal and high R² scores (0.91 and 0.92). The 24-hour forecast shows slightly more dispersed predictions, followed by the 72-hour forecast, while the 48-hour forecast shows the tightest clustering, highlighting the best performance among all options.

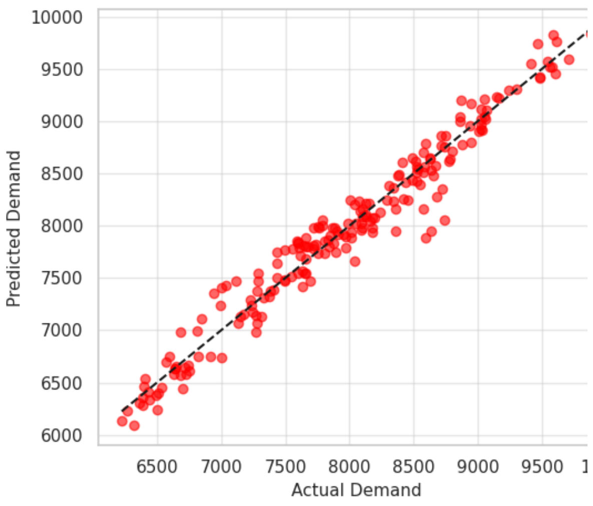
Window size: **24-hour** (128 nodes, 0 dropout, batch 32)

MAPE	MAE	RMSE	R^2
3.45	263.35	367.93	0.91

Window size: 48-**hour** (128 nodes, 0 dropout, batch 32)

MAPE	MAE	RMSE	R^2
3.23	246.82	346.83	0.92





Window size: **72-hour** (128 nodes, 0 dropout, batch 32)

MAPE	MAE	RMSE	R^2
3.22	246.53	344.01	0.92

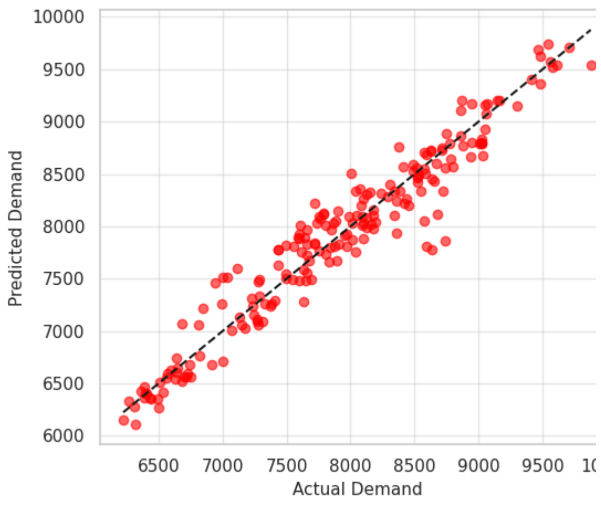


Figure 20: Actual vs Predicted Electricity Demand (First 200 Samples)

The plots below show that predictions in general follow the overall trend of electricity demand on 48-hour window sizes, showing a similar trend to the actual values. The model captures the general demand pattern. While some dips are not predicted precisely, the model still tends to follow the trend.

Window size: **48-hour**

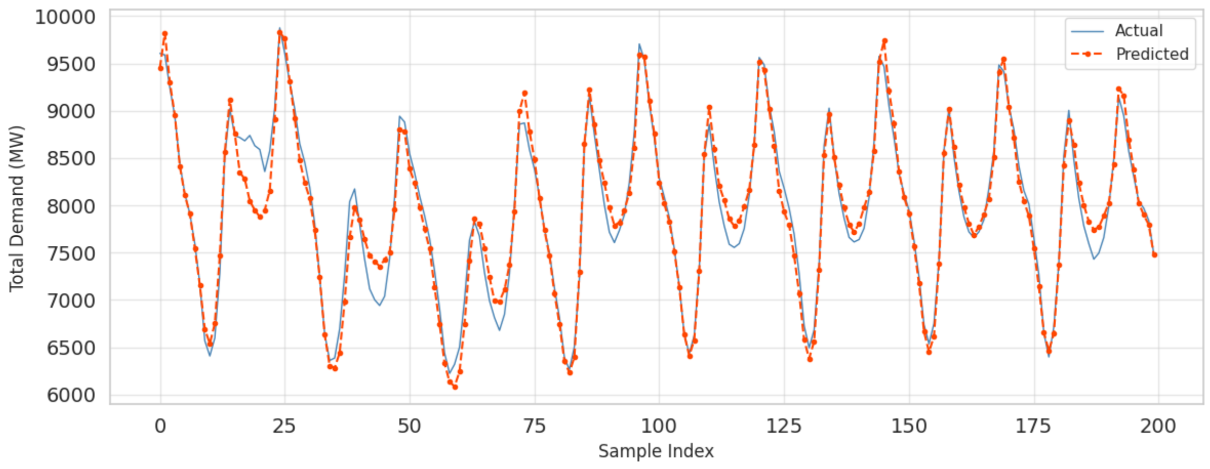


Figure 21: Predictions vs Actual - Electricity Demand (200 Random Samples)

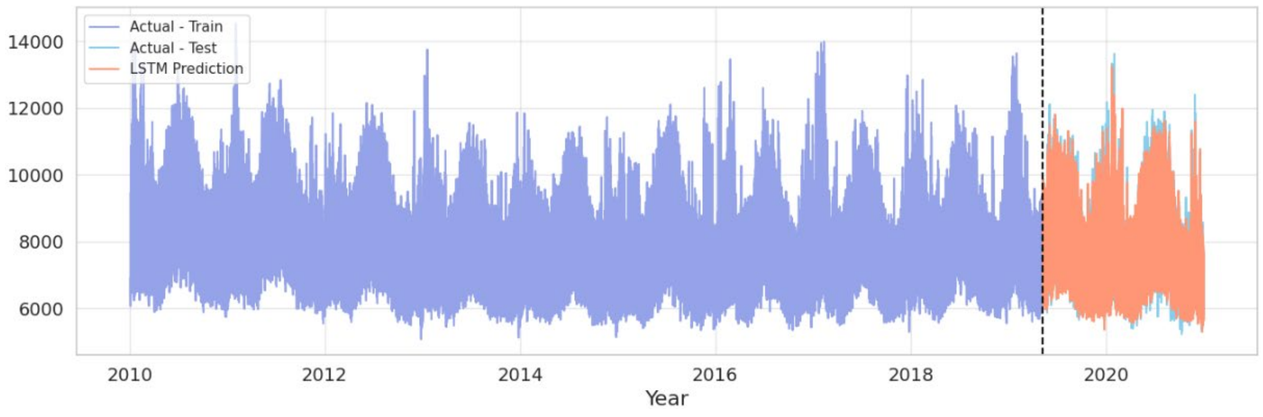


Figure 22: Predictions vs Actual - Electricity Demand (200 Random Samples)

LSTM Model Summary

Deciding on our prefered model following the analysis results was challenging, as the evaluation scores were very close, especially between 48-hour and 72-hour windows, However, the 48-hour window was identified as the most relevant strategy, ensuring the best balance between performance, accuracy and computational cost for 1-hour-ahead forecasting.

6. Discussion

One of the central concerns for the practical implementation of any machine learning model is how much end users can understand how predictions are being reached. Statistical models tend to be more interpretable than black box machine learning models. For our data set the two statistical models we worked with were outperformed by the less clear neural network. We decided that for our use case stakeholders would not be affected by the lack of transparency, but it is possible that there would be cases where the goal was to develop understanding rather than raw predictive power. In these cases, the tradeoff described may need to be reevaluated.

In our view feature engineering was the primary driver of predictive success across our models. The impact of the variables we created is visible in the outperformance of the two models which were able to include exogenous variables. For the LTSM in particular these features allowed for identification of irregular patterns. This highlights the importance of domain knowledge in model development.

We believe that there are further features that we could have included that would potentially have allowed us to develop an even more accurate predictor. The electricity market is sensitive to factors like infrastructure failures, humidity, and pricing. Incorporating these into our model would allow us a better performing model but would once again come at the cost of increasing model complexity.

The biggest constraint to the widespread application of our model is that our data uses Bankstown as a proxy for broader demand. While Bankstown shares many characteristics with wider Sydney, it may not be representative of the conditions across the rest of NSW. Applying the model outside of specifically the Bankstown grid would likely require substantial modification or allowance for substantial error.

There will be peaks and troughs in demand that are completely unpredictable. No model can be developed that would account for these. Sudden weather, sporting events, or unusual shifts in behavior can all create these sorts of impacts. While our models capture regular consumption extremely well, implementation of them into practical situations should be done in a way that includes contingencies for unforeseen events.

7. Conclusion

We set out with the goal of developing a model that could predict future demand for electricity in New South Wales. To do this we utilized 11 years of half hourly temperature and demand data from the Bankstown area. We evaluated three possible modeling methodologies, Holt Winters, Sarima, and an LTSM Neural Network.

To assist in the development of our model, we augmented the data with a number of engineered features. Cyclical transformations of time, square of temperature, typing of days, and lagged demand were incorporated into the original data set.

The Holt Winters model produced a mean absolute percentage error of 15.5% largely due to the model under- and over-estimating extremes in demand. The SARIMA model, which was able to incorporate exogenous variables such as temperature, delivered a substantially better MAPE of 5.2%. Nevertheless, the SARIMA still failed to account for the greatest peaks and troughs in demand.

The LTSM neural network benefited the most from our feature engineering. Predictions were generated from a forecasting window of either 24, 48, or 72 hours. Utilizing automated hyperparameter tuning and a series of network layers it predicted with an MAPE of 3.23% with a 48-hour training window. The longer 72-hour window produced an improvement of less than 0.01% in MAPE at substantial computational cost while the 24-hour window yielded substantially lower performance.

The main issue arising from our model selection is the trade-off of complexity and accuracy for computational efficiency and interpretability. While the two statistical methods are more easily interpretable and therefore offer greater insight into their workings for stakeholders, the performance improvement in the neural network is obvious.

It is likely that the model while good could be further developed in a number of ways. Further exogenous variables such as humidity could increase accuracy of day-to-day forecasting while more advanced machine learning techniques or computational power could rebalance the tradeoffs made between complexity and efficiency.

For real world use we believe that the 48-hour window LTSM network provides the best balance between accuracy and complexity. It is important to note that regardless of the quality of the model there will always be unpredictable events that affect demand.

No single method is able to provide all stakeholder wants. Despite this through rigorous feature engineering and model development we have developed an accurate predictor for electricity demand – supporting stakeholders at all levels in providing and accessing energy.

All code and data necessary to reproduce the results of our investigation is available at https://github.com/firefyd/Group-G-Data-Science-Project

8. References

Zhang, L., Wang, J., & Wang, X. (2017). Electricity price forecasting by a hybrid model, combining wavelet transform and kernel extreme learning machine. *Applied Energy, 190*, 291-305. https://doi.org/10.1016/j.apenergy.2016.12.157

Arslan Tuncar, E., Sağlam, Ş., & Oral, B. (2024). A review of short-term wind power generation forecasting methods in recent technological trends. *Energy Reports*. https://doi.org/10.1016/j.egyr.2024.02.140

Brailey (2024). Transforming grid operations with accurate short-term energy predictions. *DNV*. Retrieved March 31, 2025, from https://www.dnv.com/article/transforming-grid-operations-with-accurate-short-term-energy-predictions/

Australian Renewable Energy Agency. (2019). Short-term forecasting: Improving predictability of renewables. In *Australian Renewable Energy Agency Annual Report 2018–19*. Retrieved March 31, 2025, from https://www.transparency.gov.au/publications/climate-change-energy-the-environment-and-water/australian-renewable-energy-agency/australian-renewable-energy-agency-annual-report-2018-19/annual-performance-statement-2018-19/short-term-forecasting%3A-improving-predictability-of-renewables

Zhang, R., Dong, Z. Y., Xu, Y., Meng, K., & Wong, K. P. (2012). Short-term load forecasting of Australian National Electricity Market by an ensemble model of extreme learning machine. *IET Generation, Transmission & Distribution, Volume*(Issue), Page Numbers. https://doi.org/10.1049/iet-gtd.2012.0541

Román-Portabales, A., López-Nores, M., & Pazos-Arias, J. J. (2021). Systematic Review of Electricity Demand Forecast Using ANN-Based Machine Learning Algorithms. *Sensors*, 21(13), 4544. https://doi.org/10.3390/s21134544

Nti, I. K., Teimeh, M., Nyarko-Boateng, O., & Adekoya, A. F. (2020). Electricity load forecasting: A systematic review. *Journal of Electrical Systems and Information Technology*, 7(1), 13. https://doi.org/10.1186/s43067-020-00021-8

Sharifhosseini, S. M., Niknam, T., Taabodi, M. H., Aghajari, H. A., Sheybani, E., Javidi, G., & Pourbehzadi, M. (2024). Investigating intelligent forecasting and optimization in electrical power systems: A comprehensive review of techniques and applications. *Energies, 17*(21), 5385. https://doi.org/10.3390/en17215385

Taylor, J. W. (2003). Short-term electricity demand forecasting using double seasonal exponential smoothing. Journal of the Operational Research Society, 54(8), 799–805. https://doi.org/10.1057/palgrave.jors.2601589

Chujai, P., Kerdprasop, N., & Kerdprasop, K. (2013). Time series analysis of household electric consumption with ARIMA and ARMA models. In Proceedings of the international multiconference of engineers and computer scientists (Vol. 1, pp. 295-300).

Taylor, J. W. (2010). *Exponentially weighted methods for forecasting intraday time series with multiple seasonal cycles*. International Journal of Forecasting, 26(4), 627–646. https://doi.org/10.1016/j.ijforecast.2010.04.001

Waheed, W., & Qingshan, X. (2024). *An efficient load forecasting technique by using Holt-Winters and Prophet algorithms to mitigate the impact on power consumption in COVID-19*. IET Energy Systems Integration. https://doi.org/10.1049/esi2.12132

Nguyen, Q. D., Nguyen, N. A., Tran, N. T., Solanki, V. K., González Crespo, R., & Nguyen, T. N. A. (2021). *Online SARIMA applied for short-term electricity load forecasting*. Hanoi University of Science and Technology.

De Livera, A. M., Hyndman, R. J., & Snyder, R. D. (2011). Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*, 106(496), 1513–1527.

Bozkurt, Ö. Ö., Biricik, G., & Tayşi, Z. C. (2017). *Artificial neural network and SARIMA based models for power load forecasting in Turkish electricity market*. PLOS ONE, 12(4), e0175915. https://doi.org/10.1371/journal.pone.0175915

Al-Musaylh, M. S., Deo, R. C., Adamowski, J. F., & Li, Y. (2018). Short-term electricity demand forecasting with MARS, SVR and ARIMA models using aggregated demand data in Queensland, Australia. *Advanced Engineering Informatics, 35*, 1–16. https://doi.org/10.1016/j.aei.2017.10.002

Tarmanini, C., Sarma, N., Gezegin, C., & Ozgonenel, O. (2023). Short term load forecasting based on ARIMA and ANN approaches. *Energy Reports*, 9, 550–557. https://doi.org/10.1016/j.egyr.2023.01.060

Bećirović, E., & Ćosović, M. (2016). Machine learning techniques for short-term load forecasting. In *2016 4th International Symposium on Environment Friendly Energies and Applications (EFEA)* (pp. 1–4). IEEE. https://doi.org/10.1109/EFEA.2016.7748789

Darbellay, G. A., & Slama, M. (2000). Forecasting the short-term demand for electricity: Do neural networks stand a better chance?. International journal of forecasting, 16(1), 71-83.

Hadri, S., Naitmalek, Y., Najib, M., Bakhouya, M., Fakhri, Y., & Elaroussi, M. (2019). A comparative study of predictive approaches for load forecasting in smart buildings. Procedia Computer Science, 160, 173-180.

Aguilar Madrid, E., & Antonio, N. (2021). Short-term electricity load forecasting with machine learning. *Information*, 12(2), 50. https://doi.org/10.3390/info12020050

Zhao, X., Li, Q., Xue, W., Zhao, Y., Zhao, H., & Guo, S. (2022). Research on ultra-short-term load forecasting based on real-time electricity price and window-based XGBoost model. Energies, 15(19), 7367.

Arora, S., & Taylor, J. W. (2018). Rule-based autoregressive moving average models for forecasting load on special days: A case study for France. European Journal of Operational Research, 266(1), 259-268.

Fahad, M. U., & Arbab, N. (2014). Factor affecting short term load forecasting. Journal of Clean Energy Technologies, 2(4), 305-309.

Nti, I. K., Teimeh, M., Nyarko-Boateng, O., & Adekoya, A. F. (2020). Electricity load forecasting: a systematic review. Journal of Electrical Systems and Information Technology, 7, 1-19.

Wang, Y., Zhang, N., & Chen, X. (2021). A short-term residential load forecasting model based on LSTM recurrent neural network considering weather features. Energies, 14(10), 2737.

Abu-Salih, B., Wongthongtham, P., Morrison, G., Coutinho, K., Al-Okaily, M., & Huneiti, A. (2022). Short-term renewable energy consumption and generation forecasting: A case study of Western Australia. *Heliyon*, *8*(3).

Facebook (2019). *Prophet*. [online] Prophet. Available at: https://facebook.github.io/prophet/.

Zhang, G., Wei, C., Jing, C. and Wang, Y., 2022. *Short-Term Electrical Load Forecasting Based on Time Augmented Transformer*. Neural Computing and Applications, Accessed: 5 Apr 2025, https://doi.org/10.1007/s00521-022-07888-4

Chapagain, K., Gurung, S., Kulthanavit, P. and Kittipiyakul, S., 2023. *Short-Term Electricity Demand Forecasting Using Deep Neural Networks: An Analysis for Thai Data*. **Applied System Innovation**, 6(6), p.100, Accessed: 5 Apr 2025, https://doi.org/10.3390/asi6060100

Mahjoub, S., Chrifi-Alaoui, L., Marhic, B. and Delahoche, L., 2022. *Predicting energy consumption using LSTM, multi-layer GRU and Drop-GRU neural networks*. Sensors, 22(11), p.4062, Accessed: 5 Apr 2025, https://doi.org/10.3390/s22114062

Géron, A., 2022. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. 3rd ed. Sebastopol, CA: O'Reilly Media.

Or, B., 2020. *The Exploding and Vanishing Gradients Problem in Time Series*. [online] metaor.ai, Accessed: 12 Apr 2025, https://medium.com/metaor-artificial-intelligence/the-exploding-and-vanishing-gradients-problem-in-time-series-6b87d558d22

Wang, X., Huang, T., Zhu, K. and Zhao, X., 2022. *LSTM-Based Broad Learning System for Remaining Useful Life Prediction*. Mathematics, 10(12), p.2066, Accessed: 5 Apr 2025, https://doi.org/10.3390/math10122066

TensorFlow. (n.d.). *The Sequential model | TensorFlow Core*. [online] Available at: https://www.tensorflow.org/guide/keras/sequential_model.

9. Appendices

Appendix A: Feature Engineering Details

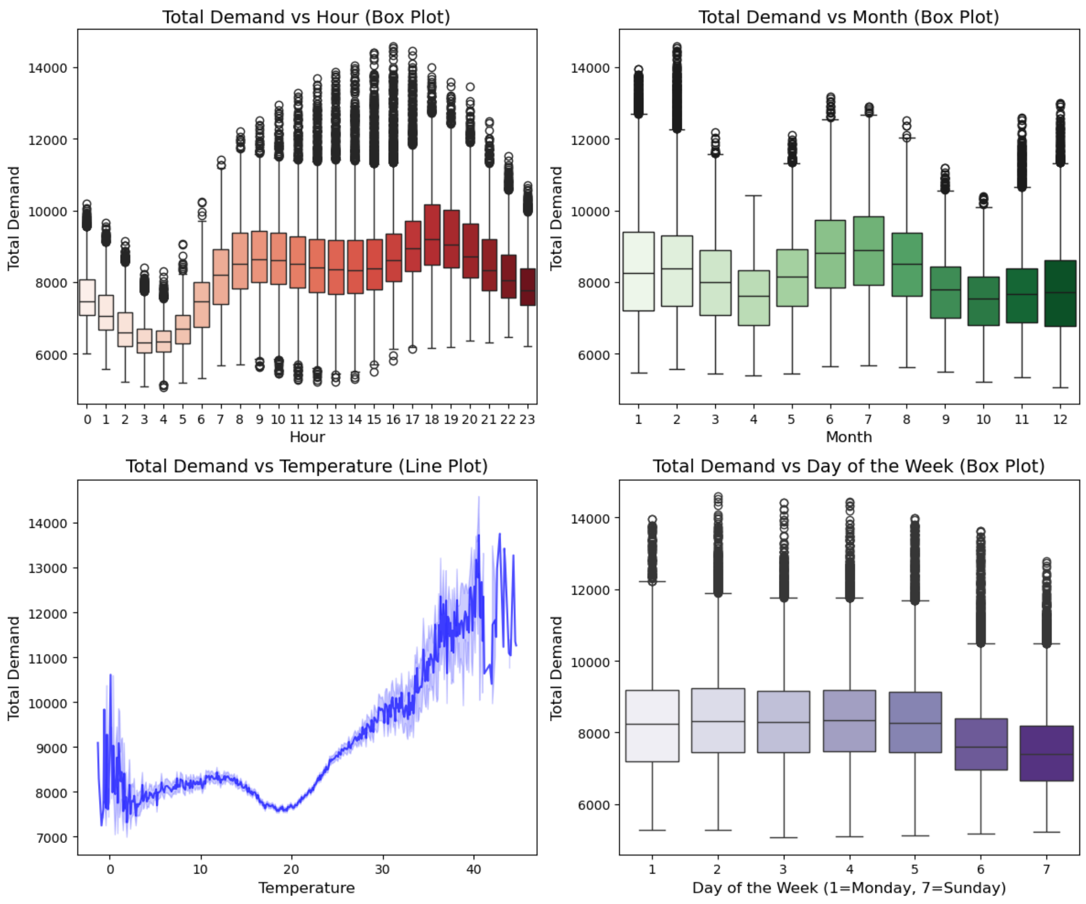
Table 1. Description of Features Used in Modeling

Features Category	Name	Type	Description & Format
Time-Based	MONTH	Numerical	Numeric representation of the month (e.g. 1 = January, ..., 12 = December)
	WEEKOFYEAR	Numerical	Week of the year (1–52 or 53)
	DAYOFWEEK	Numerical	Day of the week (0 = Monday, ..., 6 = Sunday)
	HOUR	Numerical	Hour of the day (0–23)
Weather	TEMPERATURE	Numerical	Temperature of the given date time
	TEMP_SQUARED	Numerical	Squared temperature to capture the nonlinear relationship between Temp and Demand
Lag (Time Series Memory)	LAG_24H	Numerical	Actual demand 24 hours before the current timestamp
	LAG_48H	Numerical	Actual demand 48 hours before the current timestamp
Calendar/Event	IS_WEEKEND	Bool	True if Saturday or Sunday, False for other weekdays
	IS_PUBLIC_HOLIDAY	Bool	True if the date is a public holiday (including New Year Period), False for other regular days.
Seasonality (One-Hot Encoded)	SEASON_Spring	Bool	Spring season indicator flag
	SEASON_Summer	Bool	Summer season indicator flag
	SEASON_Winter	Bool	Winter season indicator flag
Time Cyclical	DAY_SIN	Numerical	Sine transformation to capture the cyclical pattern of hours in a day
	DAY_COS	Numerical	Cosine transformation for the same daily cycle, helps in forming a circular representation of time when combined with DAY_SIN
	YEAR_SIN	Numerical	Sine transformation to capture seasonal patterns over a year
	YEAR_COS	Numerical	Cosine counterpart for annual cycles. Used together with YEAR_SIN to represent the cyclical position within a year

Appendix B : Feature Importance

```
          base_feature  importance
0              DAY_SIN_    0.369075
1                 HOUR_    0.194570
2                  LAG_    0.123048
3            DAYOFWEEK_    0.103698
4           WEEKOFYEAR_    0.053884
5             YEAR_COS_    0.043488
6           TEMPERATURE_   0.039888
7           IS_WEEKEND_    0.020541
8             YEAR_SIN_    0.019217
9    IS_PUBLIC_HOLIDAY_    0.013559
10             DAY_COS_    0.009179
11               MONTH_    0.007490
12             SEASON_S    0.002363
13             SEASON_W    0.000000
```

Appendix C: Total Demand vs (Temp, Hour, Week and Month)

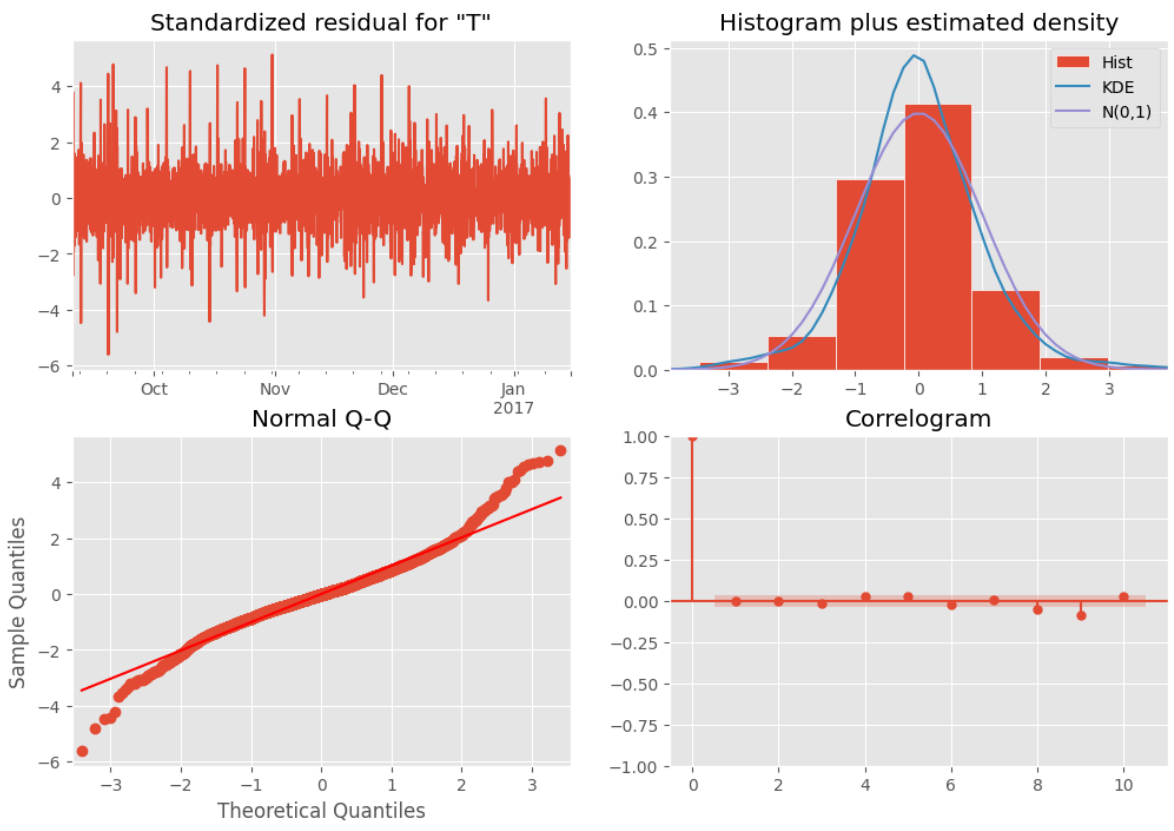


Appendix D: SARIMA Result

```
                                         SARIMAX Results
==========================================================================================
Dep. Variable:                        TOTALDEMAND   No. Observations:                 3086
Model:             SARIMAX(2, 1, 2)x(0, 1, [], 24)   Log Likelihood              -19390.093
Date:                            Sun, 13 Apr 2025   AIC                          38810.185
Time:                                    09:24:13   BIC                          38900.568
Sample:                                09-08-2016   HQIC                         38842.664
                                     - 01-15-2017
Covariance Type:                              opg
===========================================================================================
                              coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------------
TEMPERATURE                 8.0788      1.129      7.155      0.000       5.866      10.292
YEAR                        0.5632      5.747      0.098      0.922     -10.700      11.826
HOUR                        0.0053      0.054      0.098      0.922      -0.101       0.111
DAY                         0.1669      0.143      1.164      0.244      -0.114       0.448
IS_WEEKEND                -18.8876     11.287     -1.673      0.094     -41.010       3.235
IS_SEASONAL_PEAK_HOUR      40.6112     20.472      1.984      0.047       0.486      80.736
IS_SEASONAL_OFFPEAK_HOUR   92.0589     17.878      5.149      0.000      57.018     127.099
TIME_BLOCK                  0.0004      0.004      0.098      0.922      -0.008       0.008
IS_COMFORT_ZONE           -17.9046      5.736     -3.121      0.002     -29.148      -6.662
TOTALDEMANDLAG1             0.8920      0.012     73.081      0.000       0.868       0.916
ar.L1                       0.5273      0.204      2.581      0.010       0.127       0.928
ar.L2                      -0.0419      0.122     -0.344      0.731      -0.281       0.197
ma.L1                      -0.9174      0.222     -4.129      0.000      -1.353      -0.482
ma.L2                      -0.0826      0.210     -0.393      0.694      -0.494       0.329
sigma2                   1.835e+04   1450.747     12.650      0.000    1.55e+04    2.12e+04
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):              1004.83
Prob(Q):                              0.99   Prob(JB):                         0.00
Heteroskedasticity (H):               0.85   Skew:                             0.21
Prob(H) (two-sided):                  0.01   Kurtosis:                         5.78
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number 5.23e+24. Standard errors may be unstable.
```

Appendix D: SARIMA Residual



Appendix E: LSTM Architecture and Mathematical Formulation

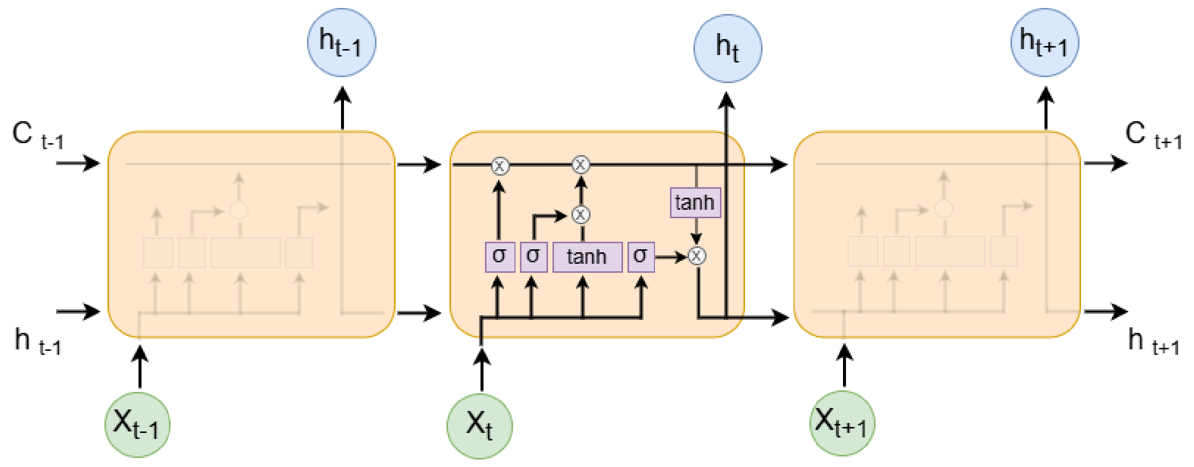


Figure 4: The repeating module in an LSTM contains four interacting layers.

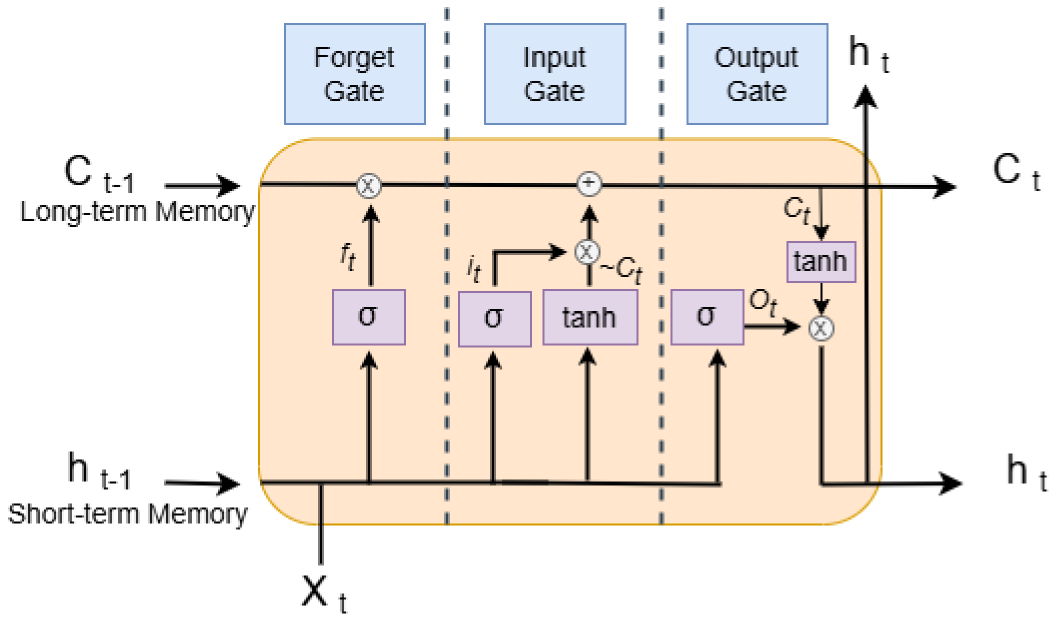


Figure 5: Structure of an LSTM cell

Forget gate

The forget gate is responsible for determining what information to remove from the cell state.The upper line (Long-term Memory) in the LSTM cell represents the cell state (C) , it flows like a conveyor belt, carrying all important information. The short-term memory belt, represented by previous state (hidden state h_{t-1}) along with Current input (X_1) are passed through a **sigmoid activation function**:

$$f_t = \sigma(W_f . [h_{t-1}, X_t] + b_f)$$

Where (W_f , b_f) are the weights and bias of forget gate

The output from f_t $\in [0 , 1]$ how much of the previous cell state C_{t-1} should be keep or forgot. Then the output from f_t multiplied with C_{t-1}

$$C_t = f_t . C_{t-1}$$

Input gate

The input gate is responsible for assigning importance for new data and what part of information should be stored in the cell state, it includes two steps:

1st step: h_{t-1}along with Current input X_1 are passed through the **sigmoid activation function**.

$$i_t = \sigma (W_i . [h_{t-1}, X_t] + b_i)$$

Where (W_i, b_i) are the weights and bias of input gate

The output from *i_t* determines what part of new information should be added to the cell state

2nd step: h_{t-1}along with Current input X_1 are passed through **tanh activation function:**

$$\tilde{C}_t = \tanh(W_c . [h_{t-1}, X_t] + b_c)$$

where (W_c, b_c) are the weights and bias of tanh

The output from $\tilde{C}_t$ represents the new information that could be added to the cell state

Then, the multiplication of the input gate output i_t and the candidate memory $\tilde{C}_t$*:*

Determine what new information to add. *(* i_t . $\tilde{C}_t$) the output then added to the updated cell state from the forget gate (f_t . C_{t-1})

The final new cell state is represented by: ***(the result of forget gate + the result of input gate)***

$$C_t = f_t \, . C_{t-1} + i_t \, . \tilde{C}_t$$

Output gate

The output gate is responsible for determining what information to output from the SLTM cell based on the updated Current state C_t

$$O_t = \sigma \, (W_o \, . \, [h_{t-1} . X_t] + b_o \,)$$

The Final Output (hidden state)

$$h_t = O_t \, . \, \tanh \, (C_t)$$

The mathematical references (Wang, Huang, Zhu and Zhao, 2022) and (Nagarale and Patil, 2023)

Appendix F: LSTM Architecture, Training Results, and Hyperparameter Summary

Window Size: 24-hour Results

Table 3. The table presents the LSTM forecast performance for the 1-hour forecast horizon using a **24-hour** window and two stacked LSTM layers. It summarizes the evaluation metrics (MAPE, MAE, RMSE, and R²) across different hyperparameter settings, highlighting the best performing hyperparameter combination set.

Hyperparameters			LSTM Results			
Nodes #	Dropout	Batch size	MAPE (%)	MAE	RMSE	R^2
64	0	32	3.81	289.64	405.53	0.89
64	0	64	3.95	299.10	404.74	0.89
64	0.2	32	4.25	327.30	462.69	0.86
64	0.2	64	4.30	328.17	468.56	0.86
128	**0**	**32**	**3.45**	**263.35**	**367.93**	**0.91**
128	0	64	3.62	274.89	378.50	0.91
128	0.2	32	4.14	314.39	444.37	0.87
128	0.2	64	4.40	337.55	475.16	0.85

Window Size: 48-hour Results

Table 4. The table presents the LSTM forecast performance for the 1-hour forecast horizon using a **48-hour** window and **two stacked LSTM layers**. It summarizes the evaluation metrics (MAPE, MAE,

RMSE, and R²) across different hyperparameter settings, highlighting the best performing hyperparameter combination set.

Hyperparameters			LSTM Results			
Nodes #	**Dropout**	**Batch size**	**MAPE (%)**	**MAE**	**RMSE**	**R^2**
64	0	32	3.27	250.59	349.22	0.92
64	0	64	3.38	257.76	360.37	0.91
64	0.2	32	3.78	289.66	415.96	0.89
64	0.2	64	3.88	296.86	422.80	0.88
128	**0**	**32**	**3.23**	**246.82**	**346.83**	**0.92**
128	0	64	3.34	253.45	353.17	0.92
128	0.2	32	3.90	298.63	424.52	0.88
128	0.2	64	3.76	287.03	411.38	0.89

Window Size: 72-hour Results

Table 5. The table presents the LSTM forecast performance for the 1-hour forecast horizon using a **72-hour** window and **two stacked LSTM layers**. It summarizes the evaluation metrics (MAPE, MAE, RMSE, and R²) across different hyperparameter settings, highlighting the best performing hyperparameter combination set.

Hyperparameters			LSTM Results			
Nodes #	**Dropout**	**Batch size**	**MAPE (%)**	**MAE**	**RMSE**	**R^2**
64	0	32	3.34	256.51	359.97	0.91
64	0	64	3.51	270.58	377.21	0.91
64	0.2	32	3.76	288.01	415.72	0.89
64	0.2	64	3.99	304.88	439.37	0.87
128	**0**	**32**	**3.19**	**244.17**	**338.67**	**0.92**
128	0	64	3.22	246.53	344.01	0.92
128	0.2	32	3.73	285.75	410.17	0.89
128	0.2	64	3.99	303.53	434.28	0.88