CAPSTONE PROJECT BY TEAM 3

A DATA SCIENCE APPROACH TO FORECAST SHORT-TERM ELECTRICITY DEMAND IN NSW

Karim Adham (z5468227)

Rishantha Rajakaruna (z5441528)

Rushmila Islam (z5456038)

School of Mathematics and Statistics
UNSW Sydney

October 2024

Submitted in partial fulfilment of the requirements of the capstone course ZZSC9020

Plagiarism statement

I declare that this thesis is my own work, except where acknowledged, and has not been submitted for academic credit elsewhere.

I acknowledge that the assessor of this thesis may, for the purpose of assessing it:

- Reproduce it and provide a copy to another member of the University; and/or,
- Communicate a copy of it to a plagiarism checking service (which may then retain a copy of it on its database for the purpose of future plagiarism checking).

I certify that I have read and understood the University Rules in respect of Student Academic Misconduct, and am aware of any potential plagiarism penalties which may apply.

By signing this declaration I am agreeing to the statements and conditions above.

Signed: _____ Date: _____

02/10/2024.

Abstract

Accurate short-term electricity demand forecasting is essential for optimizing grid operations and reducing energy costs. This research focuses on predicting electricity demand in New South Wales (NSW), Australia, by employing both the XGBoost and Prophet algorithms. These models incorporate key factors such as temperature, lagged demands, and weekday variations to capture dynamic consumption patterns. XGBoost is particularly effective in modeling complex, non-linear relationships, while Prophet excels in capturing seasonality and trends in time series data. By leveraging these complementary approaches, this study aims to improve forecast accuracy, enabling grid operators to better balance supply and demand. This not only enhances grid reliability but also minimizes the risks of overproduction or load shedding, ultimately leading to significant cost savings for both energy providers and consumers. Accurate demand forecasts allow for more efficient energy purchases and resource allocation, contributing to a more cost-effective and sustainable energy system in NSW.

Contents

Chapter 1

Introduction

Accurate electricity demand forecasting is a fundamental decision-making component for governments, regulatory bodies and businesses. We need to understand the drivers of demand fluctuation and corresponding relationships to forecast electricity demand. Weather changes, long-term climate change and macroeconomic influences such as population and economic growth are vital factors influencing demand [AEMO(2024)]. Additionally, the generation of electricity through renewable sources such as solar and wind, which doubled over the last decade, has added more complexity to forecasting due to its dependency on weather[Department of Climate Change and Water(2023)]. Electricity consumption is forecasted to increase as electric vehicle charging and broader electrification begin to impact electricity demand significantly. In 2019–20, renewable energy sources accounted for approximately 19% of the state's total electricity generation, a significant increase from past years [NSW EPA(2021)].

Demand forecasting varies from short to medium and long-term. The parameters that influence each period type also vary. Short-term forecasting focuses on small time intervals, i.e., a few minutes to days. It can be heavily influenced by daily weather, time of day, and holidays. In the case of medium to long-term forecasts, the timeframe could vary from a few months to years into the future and be determined by long-term factors such as climate change, population growth, and government policy, to name a few.

This research focuses on **short-term** forecasting. Therefore, we primarily use half hourly temperature, historical half hourly actual demand, calendar information (holidays, weekends) and time of the day as critical data sources. We analyse the relationship between demand and the features above in detail.We use two models, **XGBoost** and **Prophet by Facebook** to forecast half hourly daily demand. By using multiple metrics, we compare the performance of each model. We also compare the results of the two models against the short-term demand forecast published by the Australian Energy Market Operator.

The findings will assist market participants in making informed decisions about short-term electricity demand forecasting and contribute to a more sustainable and efficient energy landscape.

Chapter 2

Literature Review

2.1 Background

Electricity demand forecasting ensures efficient, reliable and cost-effective power operation systems. Accurate forecasts enable grid operators to balance supply and demand in real time, preventing costly overproduction or dangerous shortages that could lead to blackouts. It also helps optimise the power generation scheduling for different sectors, reducing operational costs and enhancing system reliability. As renewable energy sources like solar and wind become more integrated, forecasting plays a vital role in managing fluctuations in supply. In deregulated markets, precise demand predictions allow energy providers to make informed trading decisions, ultimately benefiting suppliers and consumers with more stable and affordable electricity

According to [NSW EPA(2021)] electricity demand from the NSW grid, it is projected to experience a slight decline over the next five years before rising more. As stated in the article, the main reason for the decrease in energy consumption is the lower consumption by the NSW industrial sector, particularly the manufacturing industry. Also, improvements in the energy efficiency of appliances and machinery helped decrease electricity demand. In addition, the increasing adoption of rooftop solar panels and battery storage systems is anticipated to further lower residential demand on the electricity grid. Accurate demand prediction is critical for ensuring optimal resource allocation and cost management, mainly as the state integrates more renewable energy into its supply mix.

2.2 Factors Affecting Load Forecasting

Load forecasting usually predicts hourly, daily, weekly, and annual values of the system demand and peak demand. Such forecasts are categorised as short-term, medium-term and long-term, depending on the time horizon. Regarding forecasting outputs, load forecasts can also be classified as point forecasts (i.e., forecasts of the mean or median of the future demand distribution) and density forecasts (i.e., estimates of the full probability distributions of the possible future demand values). As stated in [NSW EPA(2021)] the driving factors of electricity consumption and demand, forecasts can be split into two different types: —1) structural, like population growth, economic condition, electricity price, energy efficiency etc., and 2) non-structural or random, like weather condition e.g., air temperature, extreme heat or cold, bushfire, flood, etc., building type e.g., multi-storey or free-standing house, and adoption of electric vehicles and contributions of renewable energy in the power grid, etc. Many factors drive consumers to make similar choices regarding electricity consumption.

The demand is different on weekdays, public holidays, and weekends due to weather, the use of heating and cooling appliances, and many other societal factors, such as whether the beach is pleasant or if retail promotions occur. Temperature is a crucial factor, as it directly influences electricity consumption patterns. Extreme temperatures, whether very hot or cold, can lead to higher demand for heating or cooling, respectively. Forecasts often need to account for temperature variations to predict demand more accurately, especially during seasonal extremes or unusual weather patterns.

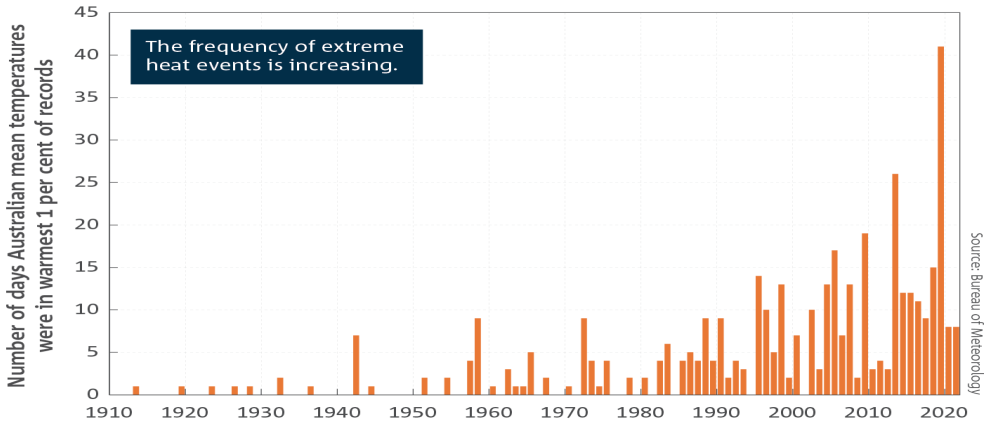

Figure 2.1: The frequency of extreme heat events in Australia from 2010 to 2020

Figure 2.1 shows the overall temperature rise in Australia, and the forecast indicates that it will continue to rise. The climate of NSW is changing, with 6 of the ten warmest years on record occurring in the past ten years. [Adapt NSW(2024)]. The warmest year on record in NSW was 2019, with an average temperature of 1.2°C above the 1990–2009 average. Across NSW, average temperatures will continue to increase throughout this century. By 2090, average temperatures will rise by around 1.3°C under a low emissions scenario and around 4.0°C under a high emissions scenario. So, temperature is critical in determining electricity demand, as heating and cooling usage fluctuations drive consumption patterns. Colder nights lead to increased heating, while hot days cause a spike in cooling system usage. These variations directly impact electricity demand, making accurate short-term forecasts crucial for grid operators to manage supply and demand effectively.

Holidays and weekdays are also critical factors in load forecasting. On holidays, electricity consumption patterns can differ significantly from regular weekdays due to changes in work routines and social activity. For instance, public holidays might see a decrease in commercial and industrial electricity use, while residential usage could increase due to family gatherings and home activities. Incorporating temperature data and considering holiday effects are essential for creating more accurate and reliable electricity demand forecasts, supporting better power system management and operational planning decision-making.

2.3 Electricity Demand Forecast Methodologies

Electricity demand or load forecasting is a well-known problem that involves predicting future load based on historical information. Econometric models and time series models are the most commonly used classical load forecasting techniques. These approaches rely on past observations to project future demand. As mentioned in [Ahmad et al.(2022)Ahmad, Ghadi, Adnan and Ali], the econometric approach combines statistical techniques with economic theory to estimate the relationship between influencing factors and energy consumption, enabling a deeper understanding of how various factors - such as economic conditions and weather - impact electricity demand across industrial, residential, and commercial sectors. The econometric approach uses historical data to provide comprehensive insights into future trends and their reasons. Despite its advantages, it may only partially capture the interdependence between quantity and prices, limiting its forecasting accuracy in some instances.

On the other hand, time series models focus on analysing historical load data to identify patterns and predict future values based solely on past trends. These models offer structural simplicity, relying entirely on previous observations to forecast future demand. While effective for forecasting, they do not explain cause-and-effect relationships between variables, thus limiting their ability to account for changes in underlying factors.

Researchers used statistical and probabilistic methods to achieve high accuracy with low errors to find the best model. They also considered this problem as deterministic and stochastic, including the influence of the external factors that sway the model's behaviour at different time horizons. As mentioned in [Ahmad et al.(2022)Ahmad, Ghadi, Adnan and Ali], load forecasting, based on time horizon, can be classified into four forecasting groups – VSLTF (very short-term load forecasting), STLF (short-term load forecasting), MTLF (medium-term load forecasting), and LTLF (long-term load forecasting).

- VSTLF: forecasts from minutes to hours (0-3h). It can deal with random variations in energy production and demand. It is used for real-time grid operation and control.
- STLF: is used for forecasting ahead of a few minutes to a few days. It is vital in different grid operations involving reliability and dispatch analysis. Further, it helps avoid overestimating and underestimating energy demand and thus contributes substantially to grid reliability.
- MTLF: In this method, the time scale expands from a few days to months ahead during a year. It helps with smart grid systems' maintenance, adequacy assessment, and fuel supply. Further, it plays a vital role in evaluating the financial attributes of energy systems by contributing to risk management.
- LTLF: This forecasting method involves a time scale ranging from months to even years. LTLF has been very important for production and load growth planning operations for a long time. Its significant advantage is that it can remove the effects of random fluctuations that occur in the short term and predict long-term trends.

We can classify the problem into three broad solution domains based on the required time horizons: heuristic, statistical or econometric, and probabilistic models. Time series datasets have the unique property of dependence on past events. Therefore, we cannot randomly shuffle the order of the data without affecting the trend. With such dependency, a simple regression technique for forecasting can randomly show statistical significance even if there is no correlation, thus suitable for real-world usage. The following sections will discuss the popular forecasting models used in the electricity demand forecasting domain.

2.4 Forecasting Models

Various techniques have been developed for electricity demand forecasting during the past few years. Electricity load forecasting models can be classified into parametric, semi-parametric, and non-parametric models based on their assumptions about the underlying data distribution and the flexibility in capturing relationships between variables. Parametric models rely on a fixed, predefined functional form with finite parameters. Statistical models are widely adopted for the load forecasting problem because of interpretability but are less flexible when dealing with complex or non-linear data[Fan and Hyndman(2012)].

Semi-parametric models, such as Generalized Additive Models (GAMs), combine parametric and non-parametric components. They allow for linear relationships between some variables while using smoothing functions to capture non-linear effects, offering a balance between interpretability and flexibility. A more recent and popular GAM for regression technique is Prophet, designed to handle business forecasting tasks featuring time series optimally captured hourly, daily, or weekly with ideally at least a full year of historical data [Taylor and Letham(2017)].

Finally, non-parametric models like Random Forests, SVR, k-nearest Neighbors, and ANNs can capture complex, non-linear relationships but are computationally intensive and more challenging to interpret. Machine learning techniques, including ANNs, have successfully forecasted load by modelling non-linear interactions between load and weather variables.

The table 2.1 shows the different forecasting methods and their advantages and disadvantages. Each method has its strengths and weaknesses, and the choice of model depends on the specific requirements of the forecasting problem.

2.5 Challenges to Find the Ideal Forecasting Method

This research focuses on short-term demand forecasting, an essential instrument in power system planning, operations, and control. Many operating decisions are based on load forecasts, such as dispatch scheduling of generating capacity, reliability analysis, and maintenance planning for the generators. Overestimation of electricity demand will cause a conservative operation, which may lead to overproduction or excessive energy purchases. The AEMO publication [AEMO(2022)] uses a monthly regression model based on five years of historical data to capture. This period is chosen to balance capturing recent univariate consumption trends while being long enough to capture seasonal patterns for reliable analysis. Univariate models, while simpler, often struggle to capture complex relationships and seasonal patterns in load data.

Multivariate models, incorporating additional factors like temperature, humidity, and economic indicators, can improve accuracy but introduce challenges in data

Table 2.1: Forecasting methods

Forecasting methods	**Advantages**	**Disadvantages**
Statistical methods	Uncomplicated and less computionally expensive.	Less reliablity for large and non-linear dataset.
Machine learning	Simple and can deal with large dataset.	Less reliable for large heterogenous data and produce point prediction.
Deep learning	Efficient for large non-linear dataset.	Creates point prediction, overfitting needs hyper-parameter tuning.
SVM	Overfitting is handled with regularization. NO local minima and many effective methods to solve problem.	Dependency on kernal and has slow testing process.
Time series analysis	Quick and accurate execution.	Numerical instability.
RNN	Can handle sequential data and effective in predict short term fluctuations.	Vanishing gradient problem, overfitting, slow training process.
LSTM	Can handle short-term and medium-term forecasting as it stores information in memory cell.	Overfitting, slow training process, requires significant computational resources.
GAMs	Can capture non-linear relationships between demand and influencing variables.	Less effective for long-term forecasting.
XGBoost	Efficient in handling large dataset and model complex non-linear relationships.	Requires hyper-parameter tuning.

collection, preprocessing, and model complexity, thus requiring careful feature engineering. Univariate models use only historical demand data to make future predictions. In contrast, multivariate models consider other variables, such as atmospheric variations and calendars, along with historical demand data in these studies of STLF [Wang et al.(2016)Wang, Wang, Xu, Zhang, Liu and Wang] and [Chen et al.(2015)Chen, Wang, Liu, Wang and Liu], both highlight the limitations of traditional methods and the growing interest in advanced hybrid techniques. Therefore, selecting an ideal forecasting method depends on factors such as data availability, computational resources, and the desired level of accuracy, making it a complex and ongoing research area.

[Taylor and McSharry(2009)] developed both univariate and multivariate models and stated that the univariate models had good prediction capability. In univariate time series models, the historical electricity demand data are arranged with correlated past lags to capture the demand patterns even when the data are limited. The model by [McCulloch et al.(2001)McCulloch, Tsay and Wu] resulted in improved accuracy by including the temperature as a variable, recognising that weather conditions play a crucial role in forecasting performance.

[Fan and Hyndman(2012)] proposes a semi-parametric additive models to estimate the relationships between demand and the driver variables. Specifically, the inputs for these models are calendar variables, lagged actual demand observations and historical and forecast temperature traces for one or more sites in the target power system. The proposed methodology has been used to forecast the half-hourly electricity demand for up to seven days ahead for power systems in the Australian National

Electricity Market (NEM). To achieve more efficient and accurate load forecasting [Suo et al.(2019)Suo, Song, Dou and Cui] establishes a multi-dimensional short-term load forecasting model based on XGBoost. The decision forest composed of many decision trees is the final learning model of XGBoost. It tries to correct the residual of all the previous weak learners by adding new weak learners. When these learners are combined for final prediction the accuracy will be higher than that of a single learner.

The selection of appropriate hyper-parameters of XGBoost is directly related to the model's performance, but there is no universal scientific method to determine them. To reduce the randomness and blindness, a technique of hyper-parameter optimisation is proposed based on the second search of multi-dimensional grids. Each hyper-parameters combination is attempted in a grid traversal manner. In short-term demand forecasting, faster computation is essential due to the need for real-time or near-real-time predictions. Models like XGBoost are preferred over LSTM because of their speed, ability to handle large datasets, faster training time, and higher interpretability. However, limitations in computational power can impact the selection of models and their performance. This is especially critical when dealing with complex models or large-scale data, where ensuring optimal performance within available resources becomes a trade-off between prediction accuracy and computational feasibility

2.6 Performance Metrics and Model Evaluations

Different criteria are utilised to evaluate the load forecasting techniques to check the correctness of the methods used to predict real load values. Many researchers have used statistical metrics to optimise the precision of their model, including newly developed statistical metrics such as probabilistic load forecasting metrics. Due to wide adaptation and extraordinary academic values in industry, the literature on probabilistic forecasting is still growing. The most critical static metrics used by researchers are shown below:

Table 2.2: Formula Table

Name of criteria	**Formula**
Mean Square Error (MSE)	$\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$
Root Mean Square Error (RMSE)	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$
Mean Absolute Error (MAE)	$\frac{\sum_{i=1}^{n}\lvert\hat{y}_i - y_i\rvert}{n}$
Mean Percentage Error (MAPE)	$\frac{1}{n}\sum_{i=1}^{n}\frac{(y_i - \hat{y_i})}{y_i}$

The most basic method to report performance is to perform hold-out validation. To do that, we split the data into 80/20 split for training and testing and 60/20/20 for training/validation/testing—the other approach to evaluate the model's performance using cross-validation. As discussed in [Rafferty(2023)], the traditional k-fold cross-validation technique is unsuitable for time series data because it shuffles the data randomly and does not consider the temporal structure of the data. Instead, time series cross-validation methods like forward chaining or rolling-origin cross-validation are like k-fold cross-validation but maintain the data's temporal order. In this research study, we will explore the hold-out validation, k-fold and forward chaining cross-validation techniques to evaluate the performance of the models. We will determine the MAE, RMSE, and MAPE to assess the performance of the models based on the test data.

Chapter 3

Material and Methods

3.1 Software

The primary software used for analysis and model build is Python. We have used an extensive set of Python libraries. The list of libraries used is as follows,

- holidays - Derive public holidays in NSW
- pandas - Data selection and manipulation
- numpy - Calculations and data manipulation
- statsmodels - Used for statistical tests and statistical data exploration
- matplotlib, seaborn - To create plots and visualize statistical analysis
- scipy - statistical analysis
- sklearn - Model preparation and analysis
- xgboost - Used for XGBoost model build
- prophet - Used for Facebook Prophet model build

Jupyter Notebook and RStudio were used as integrated development environments. The GitHub repository was used for code management. Teams and One Drive were used for collaboration and communication.

3.2 Description of the Data

This section provides a detailed description of the datasets used in this study.

3.2.1 Total Electricity Demand NSW

Total Electricity Demand in 30 min increments. This data is sourced from the Market Management System database, which is published by the market operator from the National Electricity Market (NEM) system and downloaded from [UNSW-ZZSC9020(2024)].

Table 3.1: File Details

Row Count	File Size (Approx)	File Format	File Name
196513	5.6 MB	CSV	totaldemand_nsw.csv

Table 3.2: File Attributes

Attributes	Description	Attribute Characteristics
DATETIME	Date and time interval of each observation. Format (dd/mm/yyyy hh:mm)	Timestamp
TOTALDEMAND	Total demand in MW	Numeric
REGIONID	Region Identifier (i.e. NSW1)	String. Categorical

3.2.2 Air Temperature NSW

NSW air temperature (as measured from the Bankstown Airport weather station). This data is sourced from the Australian Data Archive for Meteorology. Note: Unlike the total demand and forecast demand, the time interval between each observation may not be constant (i.e. half-hourly data). The literature review notes that temperature is a key driver of demand. Therefore, this dataset is critically essential for the research question

Table 3.3: File Details

Row Count	File Size (Approx)	File Format	File Name
220326	6.7 MB	CSV	temperature_nsw.csv

Table 3.4: File Attributes

Attributes	Description	Attribute Characteristics
DATETIME	Date and time interval of each observation. Format (dd/mm/yyyy hh:mm)	Timestamp
TEMPERATURE	Air temperature (°C)	Numeric
LOCATION	Location of a weather station (i.e., Bankstown weather station)	String Categorical

3.2.3 NSW Calendar

Use NSW Calendar for the model build to include holidays, seasons, and weekend information.

Table 3.5: Calendar Details

Attributes	Description	Attribute Characteristics
DATETIME	Date and time interval of each observation. Generated through Python library	Timestamp
HOLIDAY	Marked 1 if public holiday in NSW, otherwise 0. Generated using 'holidays' Python library Sourced from the python library https://pypi.org/project/holidays/	Numeric
SUMMER	Marked 1 if the month is in Summer season, else 0. Use one hot encoding	Numeric
AUTUMN	Marked 1 if the month is in Autumn season, else 0 Use one hot encoding	Numeric
WINTER	Marked 1 if the month is in Winter season, else 0 Use one hot encoding	Numeric
SPRING	Marked 1 if the month is in Spring season, else 0 Use one hot encoding	Numeric
WEEKDAY	Marked 1 if day of week is between Monday to Friday, else 0	Numeric
DAYOFWEEK	Marked 1 to 7 to indicate day of week	Numeric
MONTH	Month of the Year	Numeric

3.2.4 Total Forecast Demand NSW

Forecast demand in half-hourly increments for NSW. Data is also sourced from the Market Management System database. This dataset would be valuable for validating the outcome of our model, especially for understanding its accuracy.

Table 3.6: File Details

Row Count	File Size (Approx)	File Format	File Name
10906019	722 MB	CSV	forecastdemand_nsw.csv

Table 3.7: File Attributes

Attributes	Description	Attribute Characteristics
DATETIME	Date and time interval of each observation. Format (dd/mm/yyyy hh:mm)	Timestamp
FORECASTDEMAND	Forecast demand in MW	Numeric
REGIONID	Region Identifier (i.e. NSW1)	String Categorical
PREDISTPATCHSEQNO	Unique identifier of predispatch run (YYYYMMDDPP). In energy generation, "dispatch" refers to process of sending out energy to the power grid to meet energy demand. "Predispatch" then is an estimated forecast of this amount.	String (Identifier)
PERIODID	Period count, starting from 1 for each predispatch run.	Numeric (Identifier)
LASTCHANGE	Date time interval of each update of the observation (dd/mm/yyyy hh:mm)	Timestamp

3.3 Pre-processing Steps

3.3.1 Merge and read zip files

All data files (excluding NSW Calendar) were stored in zip files. Instead of exacting the the content of the files, we read directly the zip file content for data cleaning. The only exception was the forecast demand dataset, split into two zip files. Therefore, these files had to be merged into one zip before consumption.

3.4 Data Cleaning

The following activities were done as part of data cleaning.

- Verify the data types, no of rows/columns and measure of central tendency.
- Removal of unused columns.
  - Total Electricity Demand NSW - 'REGIONID' was removed as this attribute had only one value and was not useful.
  - Air Temperature NSW - 'LOCATION' was removed as this attribute had only one value and was not useful.
  - Total Forecast Demand NSW - REGIONID, PREDISPATCHSEQNO, PERIODID and LASTCHANGED were removed as they are not used for analysis
- Removal of unused rows
  - Total Forecast Demand NSW - Removes any data older than 2018-01-01 and derive the mean for the each date + time combination.
- Verify whether NULL values exist. None found in the files.

- Validate whether duplicate rows exist
  - Total Electricity Demand NSW - None found
  - Air Temperature NSW - 13 duplicate rows were removed
- Verify whether data is missing by validating that all the dates are available between the minimum and maximum dates in the file
  - Total Electricity Demand NSW - None found
  - Air Temperature NSW - Data for three dates were missing. 2016-07-16 till 2016-07-18. No action was taken as its a small percentage and also data is too far in the past and was not relevant for our research question.
- Validate whether demand is recorded consistently for each day.
  - Total Electricity Demand NSW - It was noted that for 2021, there were only 2 months of data. Also for month of March, there was only one row. Therefore this had to be removed to ensure consistency.
  - Air Temperature NSW - Temperature readings were not restricted to 30min intervals.Therefore we verified whether a temperature reading exist for every 30min. Where temperature readings were missing, we used fill forward method to add missing values. No of readings missing were 579 which is a small percentage.
- Generate Calendar Data set. Python library *holidays* was used to identify NSW holidays. Combining a date range and the holiday dates, we created a new calender dataset. Calender was limited to the date range of demand and temperature dataset.Additional attributes such as season (spring, summer, etc), day of week, weekday were derived.

The final dataset was prepared after completing the above activities. This dataset was stored as a separate CSV file for further consumption. It should be noted that additional attributes, 'HOUR' and 'PEAK', were also included. Here, 'PEAK' is defined as '1' when the time of the day is between 7:00 AM and 10:00 PM. The time period for the peak was derived from [Blue(2024)].

3.5 Assumptions

The temperature data is limited to one location. Bankstown Airport. However, the electricity demand is measured for the entire state. As we know, the temperature varies across different locations within the state. Therefore, we assume that temperature at a single point is sufficient for the demand forecast for the entire state.

The popularity of rooftop solar panels has increased over the years, and the manufacturing landscape has changed (as noted in the literature review). The impact of these factors is not part of the analysis due to the lack of publicly available data.

3.6 Modelling Methods

Based on the literature review, this study explores two popular models—the additive model Prophet by Facebook and the machine learning model XGBoost—for short-term demand forecasting. According to our EDA, demand has high variability during peak/off-peak, weekdays, hours of the day, seasons, and, most importantly, temperature change.

Since the research question relates to short-term demand, we intend to use the latest available data instead of past data. Therefore, the dataset would be restricted to 3 years. We consider random and grid searches for hyperparameter tuning for the models. The performance metrics used for the evaluation would be MAE, RMSE, and MAPE based on the *hold-out* validation with an 80/20 data split for training and testing. We will also explore the k-fold and forward chaining cross-validation to evaluate the performance of the models.

The Fig 3.1 shows the high level model diagram for the study.

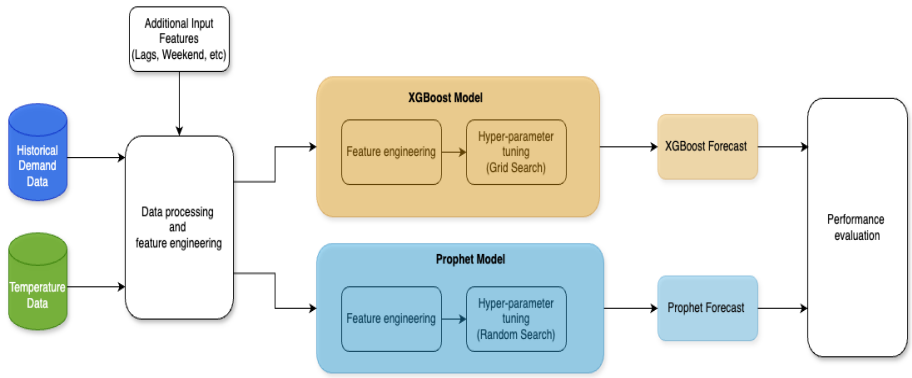

Figure 3.1: Model Diagram

3.6.1 Train Test Split

The train-test split is done differently in a time series regression model. Firstly, the chosen dataset is ordered chronologically to preserve the temporal order of the data. Eighty percent of the chronologically ordered data is used for training, and the remainder for testing [Brownlee(2020b)].

The Fig 3.2 illustrates a time series train-test split. The blue colour shows the data used for training, while the orange is the testing data. The stripped line represents the transition from training to testing.

Figure 3.2: Train-Test Split for Both Models

Chapter 4

Exploratory Data Analysis

Let's begin by analysing the data to understand its characteristics.

4.1 Yearly Electricity Demand

Analysing electricity demand over the years would help identify historical trends and any seasonal effects. First, let's review the average, minimum, and maximum demand fluctuation over the years.

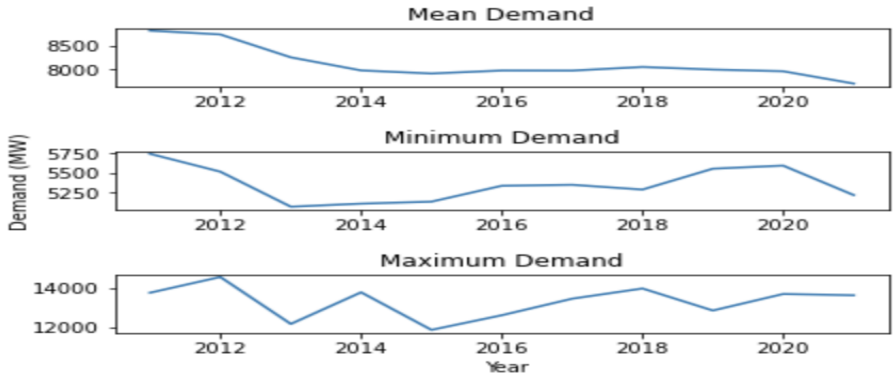

Figure 4.1: Yearly Demand Statistics

In the Figure 4.1, average demand has reduced or flat-lined over the years. We were expecting the demand to rise with population growth over the years. Therefore, using the latest available data for model build would be prudent as we focus on short-term demand. The minimum and maximum demand fluctuate within a range, indicating no significant trends.

4.2 Decompose Time series

Electricity demand data is of a time series nature. According to the Australian Energy Market Operator (AEMO), time series models are more applicable to short-term forecasting [AEMO(2023)], similar to our research question. Time series data can be decomposed to four components. [Brownlee(2017)].

- Level - The average value in the series.
- Trend - The increasing or decreasing value in the series.
- Seasonality - The repeating short-term cycle in the series.
- Noise - The random variation in the series.

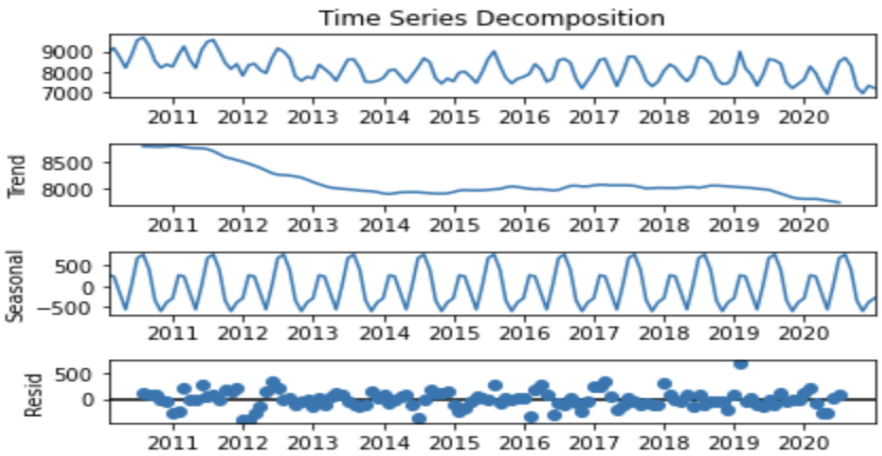

Figure 4.2: Time Series Decomposition

The Level and Trend plots both show gradual decline in demand similar to what we observed earlier. In the Figure 4.2 the seasonal plot shows the peaks and troughs in a repetitive pattern. This maybe due to relative high usage of electricity during Winter (for heating) and Summer (for cooling) compared to Spring and Autumn.

The Level and Trend plots both show gradual decline in demand similar to what we observed earlier. In the Figure 4.2 the seasonal plot shows the peaks and troughs in a repetitive pattern. This may be due to the relatively high electricity usage during Winter (for heating) and Summer (for cooling) compared to Spring and Autumn

Further, to verify that the data set used is suitable for time series analysis, we performed a stationarity test using ADF (Augmented Dickey-Fuller).

The null(H0) and alternate hypothesis(H1) of ADF test are:

- H0: The series has a unit root (value of a=1),the series is non-stationary.
- H1: The series has no unit root, the series is stationary.

If we cannot reject the null hypothesis, we can say that the series is not stationary, and if we do, it is considered stationary.

```
## Results of Dickey-Fuller Test:

## Test Statistic                  -5.586991
## p-value                          0.000001
## #Lags Used                      28.000000
## Number of Observations Used    3989.000000
## Critical Value (1%)              -3.431990
## Critical Value (5%)              -2.862265
## Critical Value (10%)             -2.567156
## dtype: float64
```

The results show that the test statistic is lower than the critical values. Therefore, we can reject the null hypothesis and conclude that the time series is stationary.

4.3 Monthly Demand

Let's analyse the impact of demand based on the month.

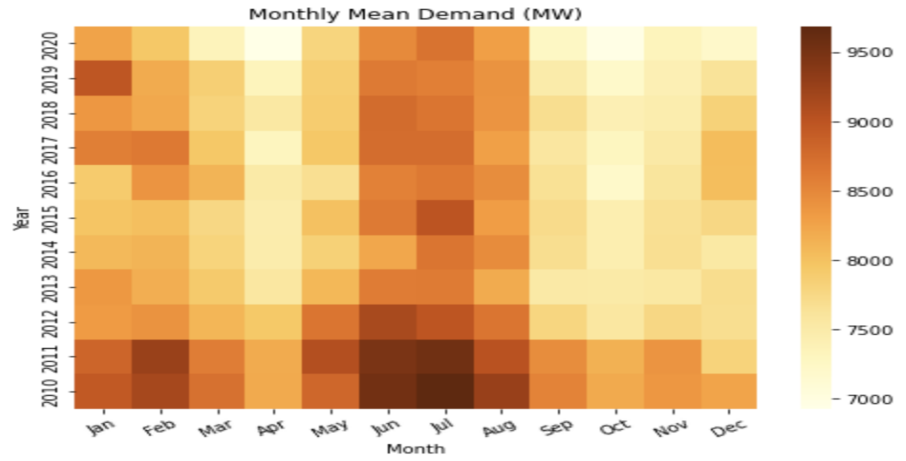

Figure 4.3: Monthly Demand Heatmap

We tend to use heating during winter and cooling during summer. The heatmap indicates that June, July, and August are winter months, and January and February are summer months. Conversely, Spring and Autumn have lower average demand. Therefore, the month/season should be considered for the model build.

4.4 Day of the week Demand

Next, we analyse whether electricity demand fluctuates depending on the day of the week. As per the heatmap Figure 4.4, weekends tend to have lower demand. This could be because most offices and factories are closed during the weekend. Also, many people tend to spend weekends outside. Similar to the month, the day of the week influences the demand. Hence, it is suitable to be included in the model build.

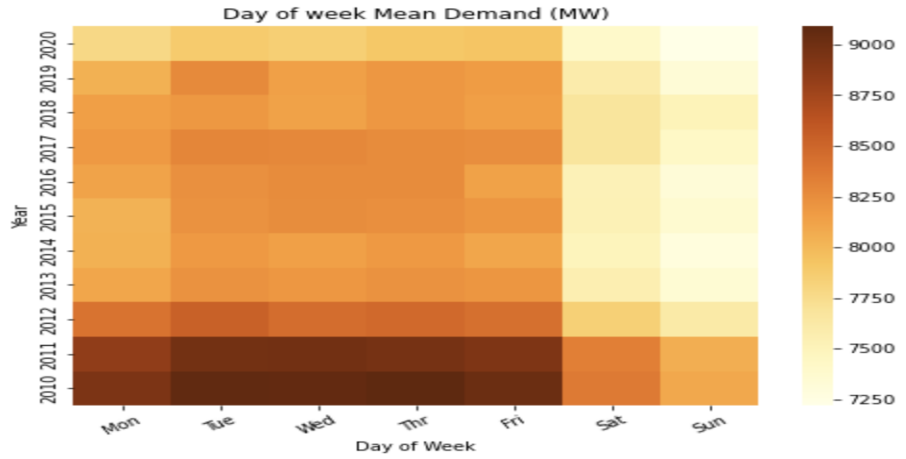

Figure 4.4: Day of the Week Demand Heatmap

4.5 Demand on Holidays

Extending on the day of the week demand, we would like to analyse the difference in average demand between Business days and non-business days. Public Holidays and weekends are considered non-business days.

The Figure 4.5 clearly indicates a significant difference in mean demand between business days and non-business days. As noted previously for Saturday and Sunday, this may be due the fact that offices, factories not operating over holidays resulting

in lower demand. Therefore we could conclude that holidays has a impact on the overall demand and therefore should be considered in the model.

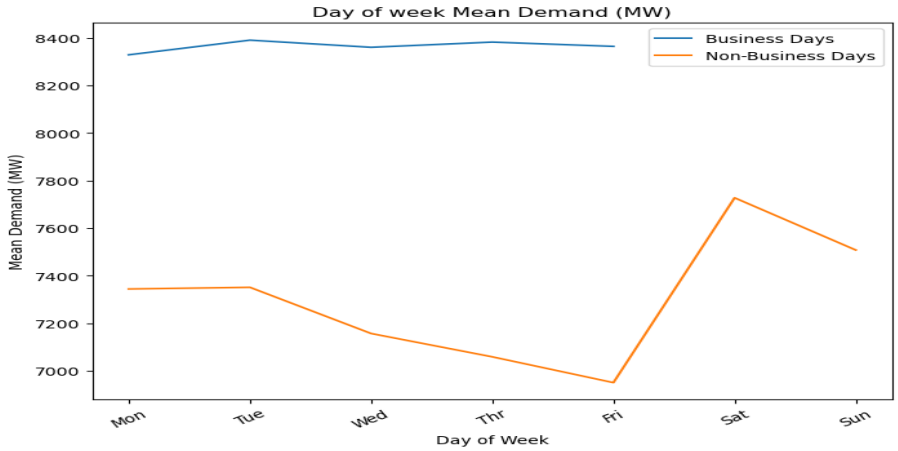

Figure 4.5: Mean Demand Business days vs Non-Business days

4.6 Hour of the day demand

At the next granular level, we would like to observe how demand fluctuates within a day (hour by hour). Australian energy providers broadly segregates hourly demand to three groups [Wrigley(2019)]. Peak, off-peak and shoulder. There are variations of this by providers. For our analysis purposes, we would simplify to Peak and Off-peak only. As evident the heatmap from the Figure 4.6, approximately from 7:00 AM to 10:00 PM, the demand seems to be high. Therefore, we would use that period as Peak demand and rest as off-peak demand for our model build.

It should also be noted that during night, demand is quite low. This is evident in Figure 4.6, the heatmap with lighter shade of yellow. Overall, we could conclude that the hour of the day is an important factor in determining electricity demand.

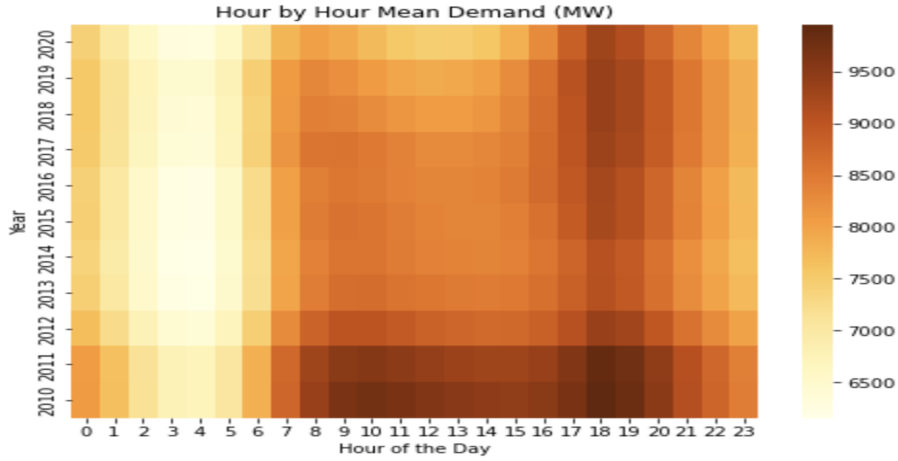

Figure 4.6: Hourly Demand Heatmap

4.7 Peak vs Off-Peak demand

Extending the previous daily demand analysis, lets verify the variations in demand during Peak hours and off-peak hours during business days vs non-business days. In

the Figure 4.7, the left side graph clearly indicate the difference in demand between peak and off-peak hours during business days. The average demand during peak hours is higher than the daily average demand. Conversely, average demand during off-peak hours is considerably lower than the daily average demand. There is a significant difference between peak hours demand compared to off-peak hours. On the other hand, the right side graph shows the peak and off-peak demand on non-business days which includes public holidays and weekends. The overall average seems to drop. However, the gap between the peak and off-peak demand continues similar to business days. The graphs further emphasise the importance of 'hour of the day' in determining demand.

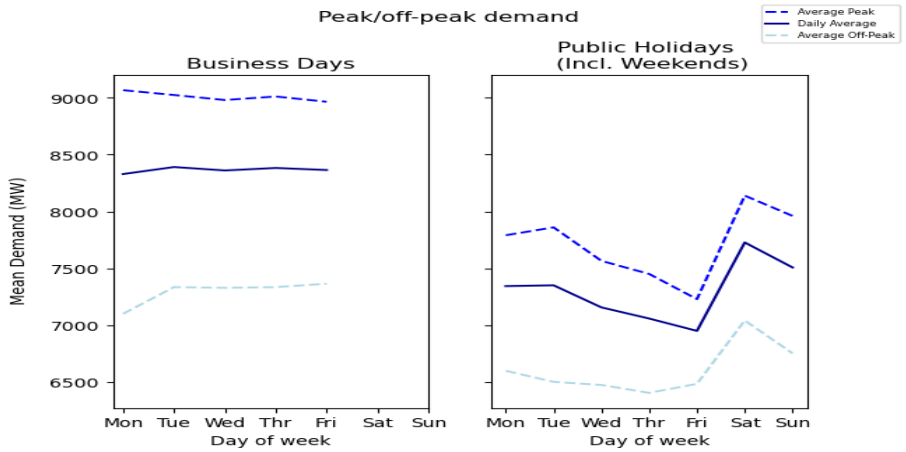

Figure 4.7: Peak vs Off-Peak Demand on Business days and Non-Business days

4.8 Autocorrelation & Lag

We would like to understand the influence of Lag in the chosen model. Prior to that lets review the concept.

Autocorrelation (serial correlation) is correlation between two values of the same variable at times t and t_k. When a value from a time series is regressed on previous values from that same time series, it is referred to as an autoregressive model. e.g y_t and y_{t-1}

$$y_t = \beta_0 + \beta_1 y_{t-1} + \epsilon_t$$

The above is a first order autoregressive model. Meaning only one proceeding value is used as predictor variable and is written as AR(1). If we used two previous values as predictors, then it would be a second order autoregressive model i.e. AR(2). This can be generalised as AR(k), i.e k^{th} order autoregressive model [Penn State University(2024)].

The autocorrelation function (ACF) is given as, $Corr(y_t, y_{t-k}), k = 1, 2, ..n$ [NIST(2020)] where k is the time gap or the lag between values of the same variable. We are interested in Partial Autocorrelation, which measure the association between y_t and y_{t-k} directly and filter out the linear influence of the random variables that lie in between. PACF is useful to identify the order of autocorrelation.

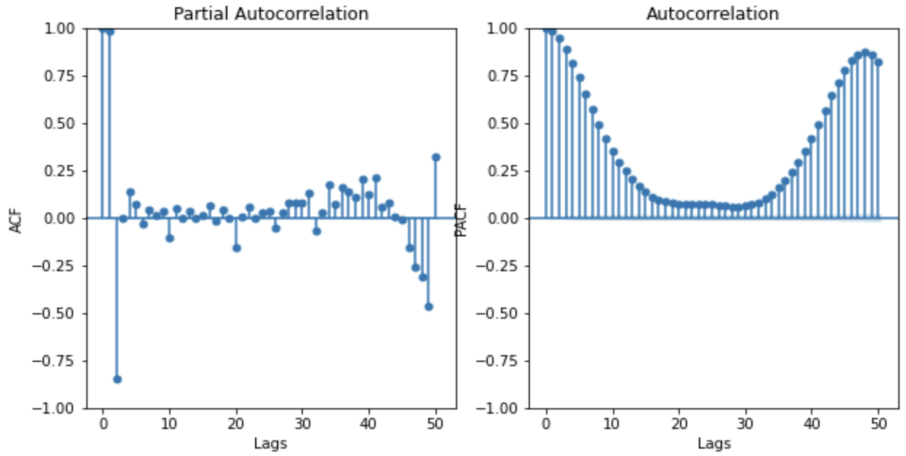

Figure 4.8: Autocorrelation and Partial Autocorrelation

The ACF plot indicates a cyclical pattern, which could be due to the seasonal effect on the demand data. It also highlights the high correlation between adjacent data points.

The PACF graph shows a significant spike in lags 1, 2, and 3 but seems to decay as it moves. Hence, it may be useful to limit the model to 3 lags.

4.9 Temperature and demand relationship

The relationship between temperature and electricity demand is well known. In the graph below, it is evident that demand increases as temperature increases. However, it is interesting to note that when temperature decreases, especially below 10 degrees, we see a limited spike in demand. A potential reason could be that in NSW, temperature falls mainly during the night / early morning, so consumers do not necessarily need heating. However, the high temperatures are primarily during the daytime, so people use electricity to cool, driving up the demand.

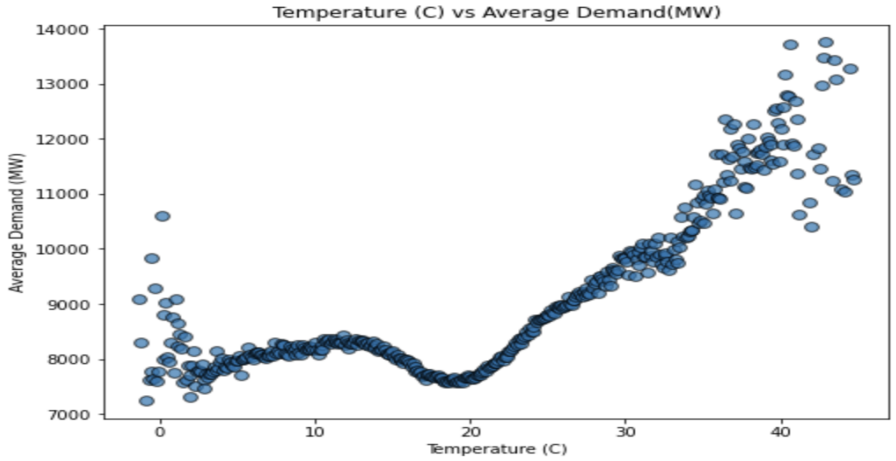

Figure 4.9: Temperature vs Demand

4.10 Correlation matrix

Finally, we look at the correlation between the attributes. The graph below shows a very low correlation between Total Demand and Temperature. This may be because the relationship is non-linear. Similarly, holidays and months have little or no

correlation with demand. In turn, hour and Peak/off-peak seem to have a higher correlation.

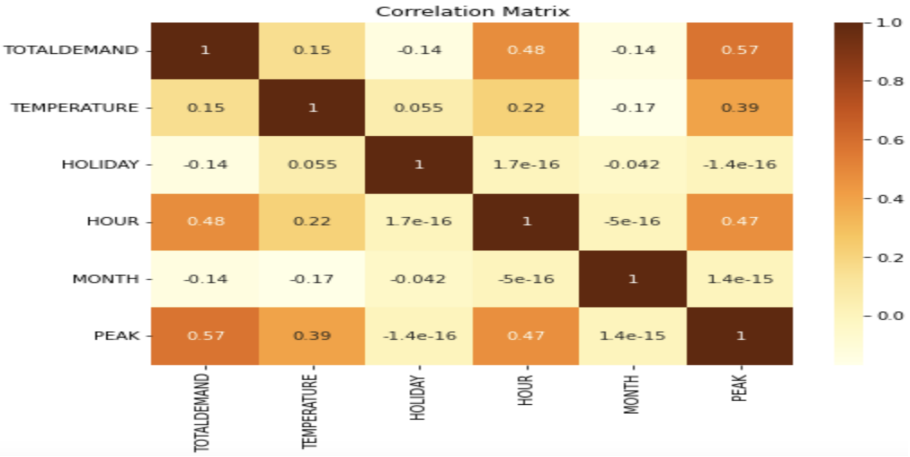

Figure 4.10: Correlation Matrix

4.11 Covid impact on Demand

Since the dataset used overlaps with the Covid period, it is essential to understand if there is any impact on overall demand.

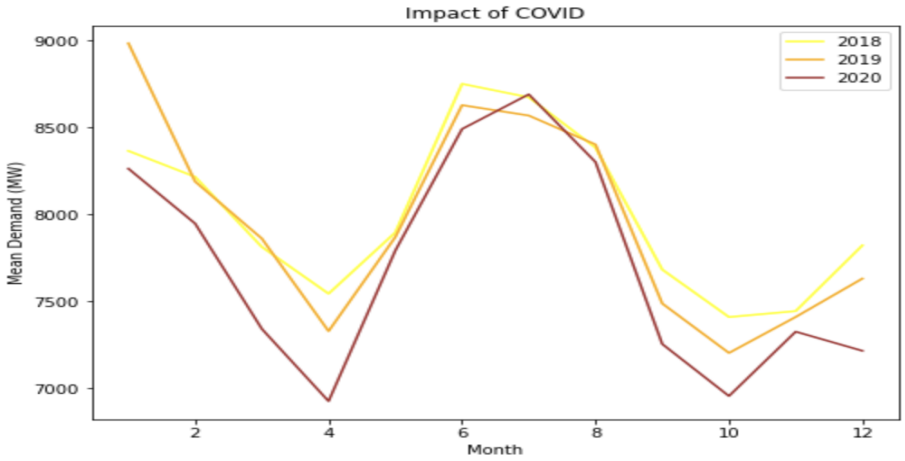

Figure 4.11: Covid impact on Demand

The plot does not indicate any significant deviations in demand. Based on the graph, one could conclude that the 2020 COVID-19 period had minimal impact on overall demand. However, further research is warranted to confirm this finding.

Chapter 5

Model Analysis and Results

5.1 XGBoost

5.1.1 Model Specification

XGBoost can be adjusted from standard regression to time series regression by applying data transformations to add lag features as predictors for the model. XGBoost is suitable for time series modelling because of its feature engineering flexibility, high accuracy, and ability to handle non-linear relationships [Brownlee(2020a)]. XGBoost's tree-based model structure can handle different features like categorical and continuous variables without normalisation or scaling like other models [Ambika(2023)]. This saves preprocessing time and creates a robust model for unscaled features.

Secondly, XGBoost can capture complex non-linear relationships because it is a decision tree-based model, and decision trees are inherently not linear [Ambika(2023)]. This allows us to use as many features as we want without worrying about feature interactions in models like linear regression or manual feature transformations like log or polynomial transformations.

Thirdly, XGBoost is highly accurate because of advanced algorithms and model-building options that make it highly effective for many machine-learning tasks [Ambika(2023)]. Combining boosting, residual learning, L1 and L2 regularisation, and pruning leads to high model accuracy. After the model is run, XGBoost shows feature importance and allows for further hyper-parameter tuning to improve accuracy and computational efficiency.

XGBoost is built upon gradient boosting, combining three elements: a loss function that must be optimised, a weak learner to make predictions, and an additive model to add weak learners to minimise the loss function. Since we are tackling a time-series regression task, the loss function is the mean Squared Error [Brownlee(2020a)].

The weak learner in XGBoost is a decision tree, but not a complex decision tree like random forests. XGBoost uses shallow trees with limited depth as weak learners. The trees are weak predictors but can be used to create an accurate model combined. An XGBoost decision tree is different from the typical decision tree. XGBoost uses the gradient and Hessian (a second-order measure of the curvature of the loss function) to improve the model, which is a further improvement over improving the model using gradient only [Chen and Guestrin(2016)].

The goal is to predict the total demand from the given features. First, each row's total demand average and residuals are calculated. Next, a similar score is calculated using the formula below. Lambda is a regularisation parameter that prevents

overfitting by penalising overly complex trees and encouraging them to split only when significant improvement occurs [Chen and Guestrin(2016)].

$$\text{Similarity Score} = \frac{\sum (residuals)^2}{\text{number of residuals} + \lambda}$$

The intuition behind the similarity score is to determine whether to split the data at a specific leaf. The goal is to determine whether splitting at a particular leaf will improve the predictive power by grouping similar residuals.

The Gain Score will be used to determine whether a split is good.

$$Gain = Similarity_{left} + Similarity_{right} - Similarity_{root}$$

The similarity score for the root and leaf nodes can be calculated, and the Gain formula can be applied. We can test different thresholds by comparing Gain from different thresholds. The higher the gain score, the better the threshold for splitting residuals into clusters of similar values. Once the best threshold for splitting is determined, the same process is applied to the leaves; to prevent overfitting, a max tree depth can be set in the model parameters. Max tree depth and lambda can both help reduce overfitting. The above process is run until the max tree depth or no more residuals are split [Esri(2016)].

The next step is to prune the trees we made. A value Gamma is chosen, and we subtract the gain of each branch by Gamma. If the result is negative, that branch doesn't provide enough improvement, so the branch is pruned. The lambda decreases the similarity score, hence decreasing the Gain value. A smaller Gain value means it is easier to prune branches and even whole trees [Esri(2016)].

Prediction formula (Sneha, 2020):

$$\text{Predicted values} = \text{Average total demand} + \text{Learning rate} \times \text{Average of each leaf}$$

For example, the initial prediction is that the average total demand is 7,000 MW, and the learning rate is 0.3. The first observation had a temperature of 20, so the average of that leaf was -25. The predicted value is:

$$7,000 + (0.3 \times (-25)) = 6,993$$

The process is repeated for each value we want to predict, and the residuals are calculated again, repeating until the maximum number of trees is reached or the model does not improve.

5.1.2 Hyperparameter Tuning

Hyperparameter tuning is critical to improving model accuracy. The two methods considered for tuning are Grid Search and Random Search. Grid Search is an exhaustive search testing all possible combinations for hyperparameter values. Random Search randomly samples a fixed number of hyperparameter combinations. The trade-off is finding the best combination of hyperparameters vs. computational efficiency. Random Search is chosen due to computational constraints.

The first hyperparameter is the number of trees (n_estimators) the model can build. A more significant number of trees allows the model to capture more complex patterns but increases the risk of overfitting and longer training time. The tuning range for the number of trees is [100,200,300,500], allowing for a gradual increase in trees for balanced exploration of a simple model with a few trees and complex models with many trees. Tree addition has diminishing returns; after a certain number of trees, only computational cost will increase, but model performance will plateau [Readthedocs(2022)].

The learning rate determines how much each tree contributes to the final prediction. A lower learning rate means the model learns slower and has a higher training time. While a significant learning rate means the model will learn fast using less computational power, it is more prone to overfitting. The small learning rate (0.01) allows the model to capture underlying patterns, but convergence is slow. The low learning rate(0.05) uses fewer trees but is still enough to prevent overfitting. This value balances run time and model accuracy [Readthedocs(2022)].

The standard learning rate (0.1) is usually used as a benchmark to compare lower or higher learning rates. The high learning rate (0.3) helps reduce training time, but with a high convergence speed, the model will have a high variance. The high learning rates are explored to determine whether faster learning provides better results or leads to overfitting. The max depth hyperparameter controls the depth of each tree. Deeper trees are better at capturing more complex patterns but have a higher risk of overfitting. The tree depth has the same rationale as the learning rate with shallow trees (3), medium depth (5), increased depth (7), and deep trees (10). The trade-offs are the same between bias, variance, overfitting, and computational efficiency [Readthedocs(2022)].

The gamma parameter (Lambda) in XGBoost is a regularisation parameter that prevents the model from overfitting. No regularisation (0) means no tree-splitting restrictions, capturing detailed patterns. No regularisation serves as a base for comparing regularisation levels. Medium regularisation (0.1) allows only significant splits, which leads to better generalisation and a balance between complexity and pruning. High regularisation (0.3) focuses on essential splits and improves model simplicity [Readthedocs(2022)].

The random search uses 50 iterations for computational efficiency. Each iteration takes a combination of random values from each hyperparameter list and tests the performance. The lowest mean squared error determines the best model. Sci-kit, by default, tries to maximise scores, so setting a negative mean squared error addresses that issue.

Cross-validation is done differently in time series. The `TimeSeriesSplit` ensures that the training happens on past data and tests on future data based on past information to preserve the temporal aspect of the time series [scikit-learn(2024)]. The best results after a random search are:

```
Best Parameters: {'n_estimators': 300,
                  'max_depth': 5,
                  'learning_rate': 0.1,
                  'gamma': 0.3}
```

5.1.3 Model Validation

The box plot in Figure 5.1 shows the mean squared error across five splits. Fold two has a higher variability in RMSE compared to other folds. There are many outliers with much higher RMSE values, which could indicate that the model struggled to maintain consistent performance. Folds one and five have tighter boxes, suggesting more stable and consistent performance in those periods.

Several outliers are present in all the folds, which might indicate that the model cannot handle specific instances well. Fold two has the most significant RMSE outlier, showcasing that the model could be quite off from actual demand. The median ranges between 100 and 200 RMSE, suggesting the overall performance across different splits is stable. The small interquartile range in folds one, three, four, and five could indicate that the model generalises well. However, fold two suggests that the model can potentially deal with high variance during specific times.

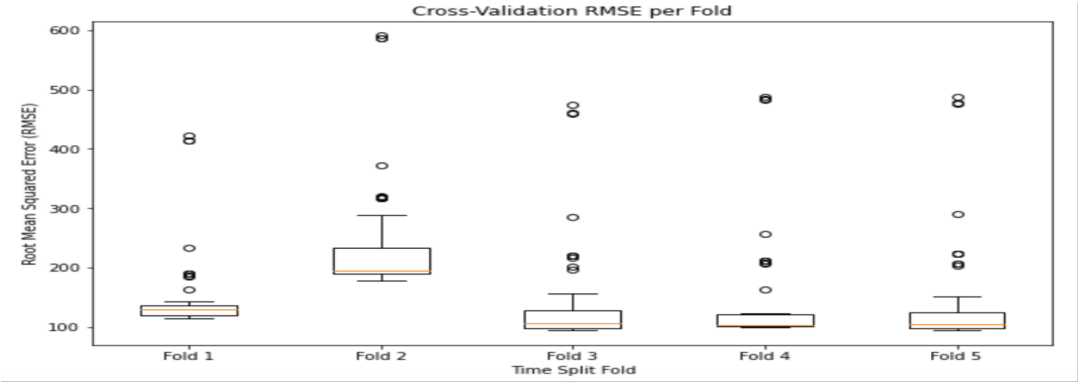

Figure 5.1: Cross Validation Performance for XGBoost Model

5.1.4 Model Outcomes

Feature Importance

The F score represents the number of times a feature is used to split data. The higher the f-score, the more critical a feature is to the model. The most essential features are Temperature, lag, and hour. The model is not too reliant on one feature, as the top three features have very close F scores. Temperature is expected to have a significant impact, as observed in the literature. Lag 1, the previous demand for the last 30 minutes, is expected to contribute significantly to a time series regression model. Three lag features provided the maximum model improvement.

The hour is transformed using the cosine function to address the issue of hour proximity [avanwyk(2022)]. For example, if we have hour "23," hour "00," and hour "05," the model might assume that hour "05" is closer to hour "00," but that is false. The cosine transformation will address this issue, giving peak hours more significance.

The hour has a significant impact and captures the effect of intraday demand. The seasons are one-hot encoded, so the interpretation differs from numerical values. The most important features are the closest to the time interval forecasted.

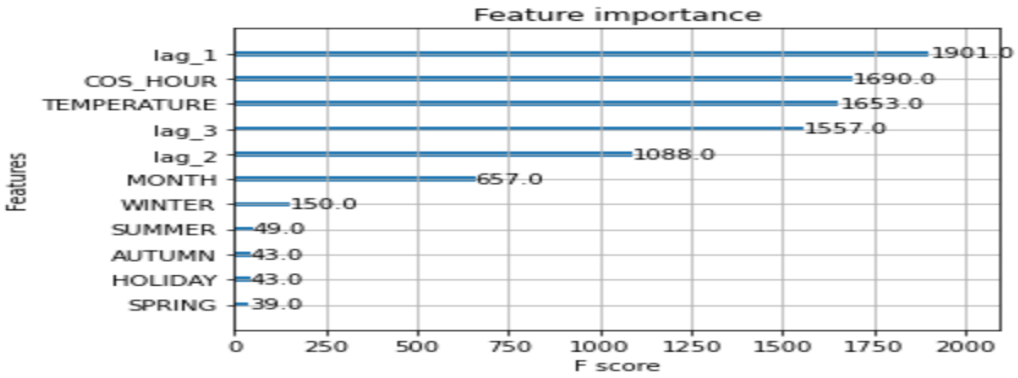

Figure 5.2: Feature Importance for XGBoost Model

Actual & Predicted Values

The Figure 5.3 shows an upward-sloping diagonal line, which indicates that the model did an excellent job predicting the target variable.

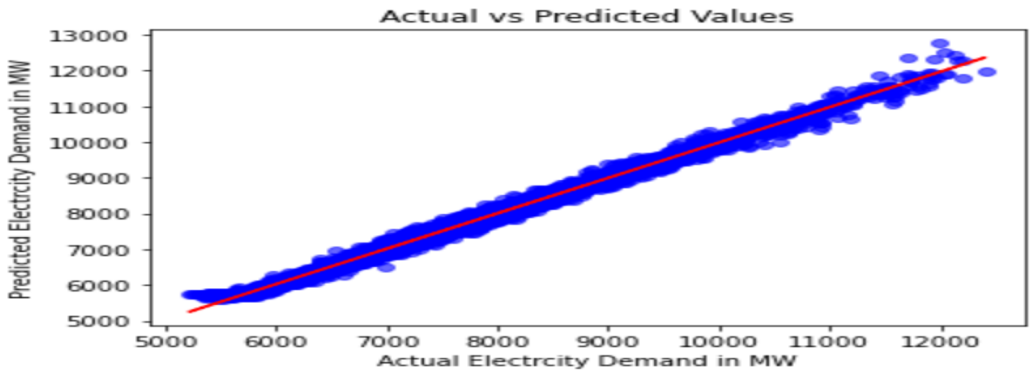

Figure 5.3: Feature Importance for XGBoost Model

Model Metrics

The three metrics used to measure performance are root mean squared error (RMSE) and mean absolute error (MAE).

```
Root Mean Squared Error : 95.43
Mean Absolute Error: 70.77
Mean Absolute Percentage Error: 1%
```

The RMSE measures the average size of the error between the actual and predicted values. An RMSE of 95.43 means that, on average, the predicted values are off by 95.43MW. The MAE measures the average of the absolute difference between the actual and predicted values. MAE is more robust against outliers because it does not square the error. An MAE of 70.86 means that, on average, the model is off by 70.77MW. The MAPE is 1% of the target range (5000 MW—13000 MW). This suggests that the model is performing reasonably well.

5.1.5 Forecasting

The last 17 days of the dataset will be forecasted using the XGBoost and Prophet models to compare with the provided AEMO forecasted demand.

```
RMSE: 88.13
MAE: 67.04
MAPE: 1.0%
```

Figure 5.4 shows the forecasted demand for the last 17 days of the dataset using the XGBoost model compared to the actual demand. XGBoost follows the general pattern of the actual demand. The model was able to predict the peaks and valleys, which are often more volatile and challenging to predict. There doesn't seem to be any prediction lag, as the model effectively captures rapid changes.

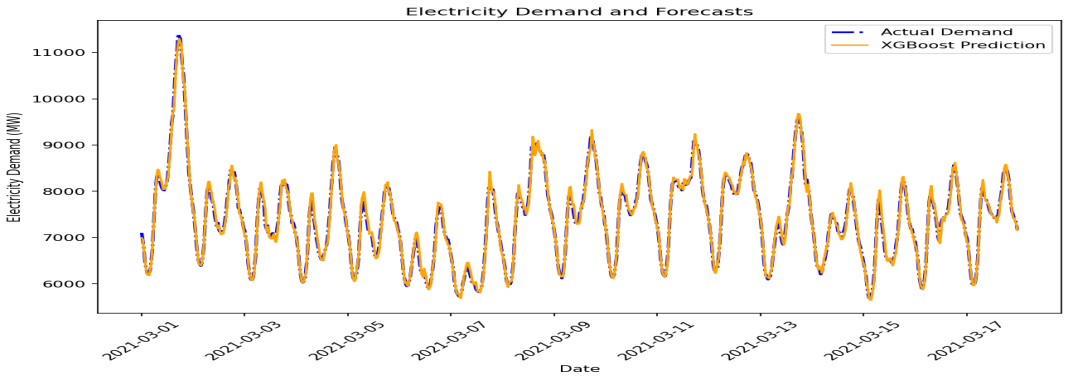

Figure 5.4: Forecast plot of XGBoost Model

5.2 Facebook Prophet

5.2.1 Model Specifications

Prophet was developed internally at Facebook (now known as Meta) to optimally handle business forecasting tasks, typically featuring multiple seasonality, missing data, and outliers [Taylor and Letham(2017)]. Prophet is a decomposable time series model with three main components: trend, seasonality, and holidays. The trend component models non-periodic changes in the value of the time series. The seasonality component models periodic changes in the value of the time series. The holiday component models the effects of holidays on potentially irregular schedules over one or more days. The model is fitted to historical data, and future data is forecasted using the model. The model is highly customisable, allowing users to adjust it to fit their data and business needs. This model is a regression model with interpretable parameters that can be intuitively adjusted by analysts with domain knowledge about the time series. The proposed Prophet model uses a decomposable time series model with three main model components: trend, seasonality, and holidays. They are combined in the following equation:

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

Here $g(t)$ is the trend function which models non-periodic changes in the value of the time series, $s(t)$ represents periodic changes (e.g., weekly and yearly seasonality), and $h(t)$ represents the effects of holidays which occur on potentially irregular schedules over one or more days. The error term represents any idiosyncratic changes which are not accommodated by the model; they make the parametric assumption that t is normally distributed.

This specification, similar to a generalised additive model (GAM), a class of regression models with potentially non-linear smoothers applied to the regressors. Here we use only time as a regressor but possibly several linear and non-linear functions of time as components. Modelling seasonality as an additive component is the same approach taken by exponential smoothing. Multiplicative seasonality, where the seasonal effect is a factor that multiplies $g(t)$, can be accomplished through a log transform.

The growth is typically modeled as a logistic growth curve, which is a common pattern in many business time series. The logistic growth model has an inflection point that can be used to model changes in growth rates. The logistic growth curve is defined as:

$$g(t) = \frac{C}{1 + \exp(-k(t - m))}$$

Where C is the carrying capacity of the growth, k is the growth rate, and m is the offset prarameter. The trend component models the non-periodic changes in the value of the time series. The model also provides a piecewise linear or logistic growth curve, which is useful for modeling growth that changes over time.

Business time series often have multi-period seasonality as a result of the human behaviours they represent. For instance, a 5-day work week can produce effects on a time series that repeat each week, while vacation schedules and school breaks can produce effects that repeat each year. To fit and forecast these effects we must specify seasonality models that are periodic functions of t. The model relies on the Fourier series to provide a flexible model of periodic effects. Let P be the regular period we expect the time series to have (e.g. P = 365.25 for yearly data or P = 7 for weekly data when we scale our time variable in days). We can approximate arbitrary smooth seasonal effects with the following:

$$s(t) = \sum_{n=1}^{N} (a_n \cos(\frac{2\pi nt}{P}) + b_n \sin(\frac{2\pi nt}{P}))$$

Seasonality is estimated using a partial Fourier sum. The number of terms in the partial sum (the order) is a parameter that determines how quickly the seasonality can change. The default Fourier order for yearly seasonality is 10, which produces this fit. The default values are often appropriate but can be increased when the seasonality needs to fit higher-frequency changes and generally be less smooth. The Fourier order can be specified for each built-in seasonality when instantiating the model. Increasing the number of Fourier terms allows the seasonality to fit faster-changing cycles but can also lead to overfitting.

Holidays and events provide large, predictable shocks to many business time series. They often do not follow a periodic pattern, so their effects are poorly modelled by a smooth cycle. As with seasonality, we use a prior $\kappa \sim N(0, \nu)$.

Prophet will, by default, fit weekly and yearly seasonality if the time series is more than two cycles long. It will also fit daily seasonality for a sub-daily time series. We can add other seasonality (monthly, quarterly, hourly) using the `add_seasonality` method.

This section will discuss implementing the Facebook Prophet model for short-term electricity demand forecasting. As discussed earlier, Facebook Prophet is a powerful tool for time series forecasting that can handle multiple types of seasonality, holidays, and missing data. We will walk through data pre-processing, model parameter tuning, and cross-validation to evaluate the model's performance.

5.2.2 Data pre-processing

Prophet follows the sklearn model API structure, where we first create an instance of the `Prophet` class and then use its `fit` and `predict` methods.

The input for Prophet must always be a DataFrame containing two specific columns: `ds` and `y`. The `ds` column (representing the date) should be in a format recognized by Pandas, such as `YYYY-MM-DD` for a date or `YYYY-MM-DD HH:MM:SS` for a timestamp. So, we have changed our `DATETIME` column to `ds`. The `y` column should be containing numeric values representing the variable we want to forecast, thus we change the `DEMAND` column to `y` column.

5.2.3 Hyper-parameter tuning

For the model prophet it is recommended to tune the parameters like - `changepoint_prior_scale`, `seasonality_prior_scale`, `holidays_prior_scale`, and `seasonality_mode`. We have considered the following parameters for tuning:

- The `seasonality_prior_scale` parameter controls the flexibility of the seasonality. Similarly, a significant value allows the seasonality to fit large fluctuations; a small value shrinks the magnitude of the seasonality. The default is 10., which applies no regularisation.
- The `holidays_prior_scale` parameter controls flexibility to fit holiday effects. to fit holiday effects. It also defaults to 10.0, which basically means no regularisation since we usually have multiple observations of holidays and can estimate their effects well. This could also be tuned to a range of [0.01, 10], as with seasonality_prior_scale, which we decide based on the grid search results discussed later.
- The `seasonality_mode` parameter options are [`'additive'`, `'multiplicative'`]. The default is `'additive'`. This is best identified by looking at our data's time series. We observe that the seasonal fluctuations are roughly constant in size over time, so we consider using additive seasonality.

Prophet does not have a built in grid search method so we have used `sklearn` the `ParameterGrid` method from the `sklearn` library to tune the hyper-parameters. We have specified the grid parameters and run the grid search. We have used the `mean_squared_error` as the scoring parameter for the grid search. The best parameters are selected based on the lowest mean squared error.

```
Best Parameters: {'daily_seasonality': True,
                  'holidays_prior_scale': 0.1,
                  'seasonality_mode': 'additive',
                  'seasonality_prior_scale': 1.0,
                  'weekly_seasonality': True,
                  'yearly_seasonality': True}
```

5.2.4 Model parameters

Our model considers holidays, so we need to create a dataframe for them. It has two columns (`holiday` and `ds`) and a row for each occurrence of the holiday. It includes all occurrences of the holiday, both in the past (back as far as the historical data go) and in the future (out as far as the forecast is being made). We have included columns `lower_window` and `upper_window` which extend the holiday out to `[lower_window, upper_window]` days around the date. For example, we wanted to include Christmas eve and Boxing day in addition to Christmas day we need to include `lower_window=-1,upper_window=1`.

Final model parameters:

```
model = Prophet(daily_seasonality = True,
                weekly_seasonality=False,
                yearly_seasonality=True,
                holidays=holiday_nsw_2018_2021,
                holidays_prior_scale=0.1,
                seasonality_prior_scale=10)
```

Additional regressors are used in the model and they are added to the linear part of the model using the `add_regressor` method. Based on the literature review, EDA and experimentation we have considered the following regressors for the Prophet model to forecast demand:

- Continuous variable: `TEMPERATURE`, `lag_1`, `lag_2`, `lag_3`
- Binary variable: `SUMMER`, `AUTUMN`, `WINTER`, `SPRING`

We these parameters in the model as additional regressors :

```
['TEMPERATURE','SUMMER', 'AUTUMN', 'WINTER', 'SPRING',
        'lag_1', 'lag_2', 'lag_3']
```

5.2.5 Cross-Validation

The traditional method to tune model's performance is the **hold-out** validation that splits the whole dataset in to training validation and test set. We have applied 80/20 split for training and testing. The model is trained on the train set and performance is evaluated on test set.

As mentioned in the literature review that we consider forward-chaining cross validation, we perform 5-fold cross validation for a range of historical cutoffs using the `cross_validation` function. We have used `cutoffs` keyword `cross_validation function` and specify the custom cutoffs.

In the Fig 5.5 we have shown the terminology used in the cross-validation method.

- 'initial' is the first training period. In Fig 5.5, it would be the first two blocks of data in the first fold. It is the minimum amount of data needed to begin the training and we need to at least have 365.6 days of data to capture the yearly seasonality. Considering our dataset we have used 420 days (14 months) as 'initial' parameter.
- 'horizon' is the length of time we want to evaluate the forecast over, in this case we have set the horizon to 15 days.

- 'period' is the amount of time between each fold. We have used approximately 90 days as the period.
- 'cutoffs' are the dates when each horizon will begin.

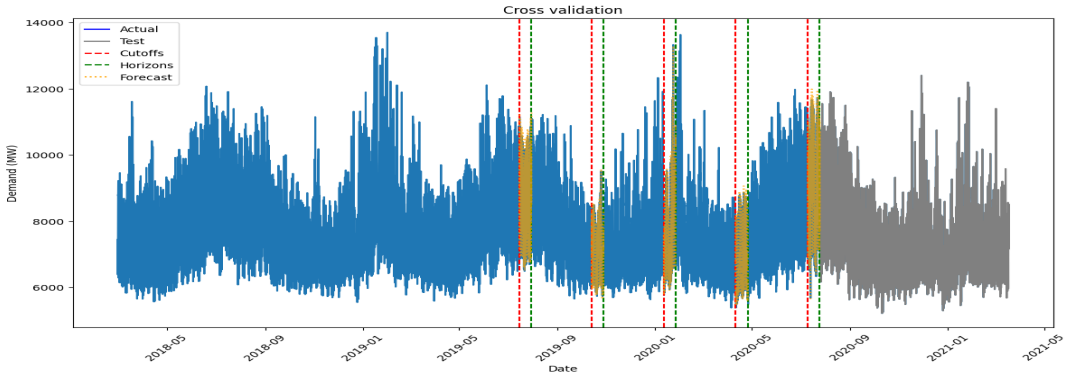

Figure 5.5: Cross-validation terminology

5.2.6 Model Results

Below is the result of the Prophet model:

```
Root Mean Squared Error: 111.57
Mean Absolute Error: 87.67
Mean Absolute Percentage Error: 0.01
```

The model has performed well in forecasting the electricity demand for the test data. The MAPE is 0.01 which is 1.0% and indicates that the model is performing well.

The result of cross validation:

Cutoff	MSE	RMSE	MAE	MAPE
2019-07-15 18:30:00	13920.91	87.61	87.61	1.00
2019-10-13 18:30:00	6465.12	63.24	63.24	0.88
2020-01-11 18:30:00	10061.93	77.71	77.71	0.96
2020-04-10 18:30:00	8314.68	66.03	66.03	0.95
2020-07-09 18:30:00	16096.51	92.95	92.95	1.04

The cross-validation results show that the model has performed well for almost all the cutoffs. The MAPE is also close to 1% for all the cutoffs which indicates that the model is performing well.

The Fig 5.6 shows the trend and seasonality components of the model. The trend component shows the overall trend of the electricity demand, while the seasonality component shows the daily and weekly seasonality patterns. The holidays component shows the impact of holidays on the electricity demand. As we have predicted the `yearly` seasonality shows the summer and winter spike.

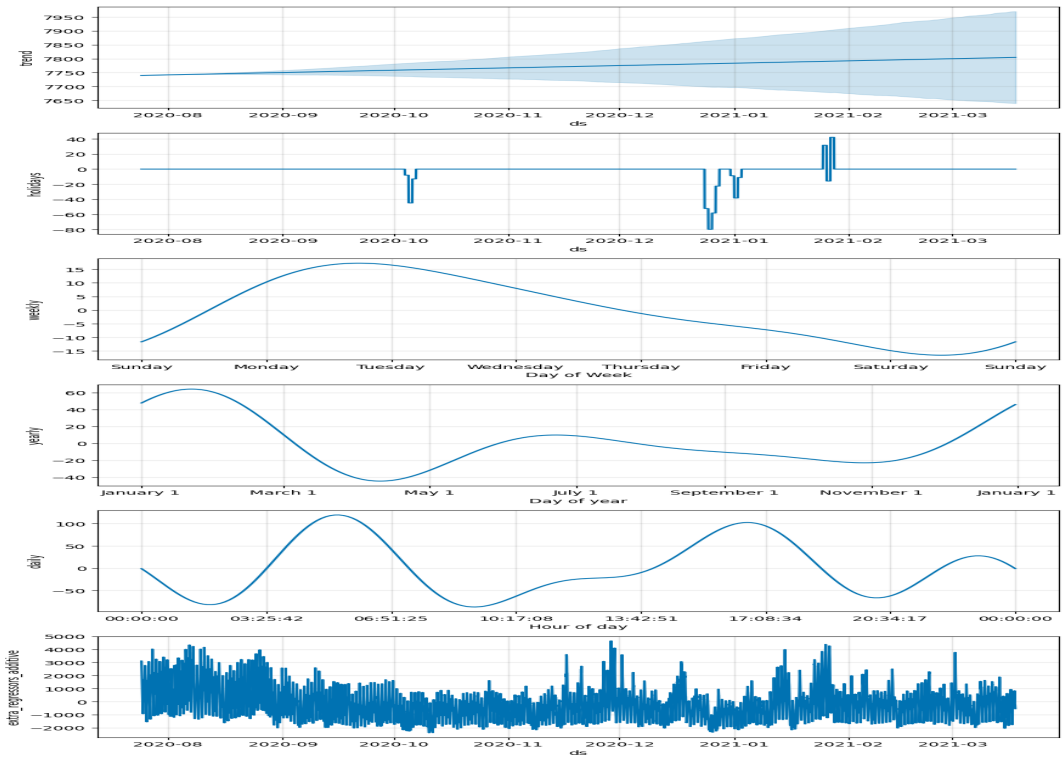

Figure 5.6: Component plots of model Prophet

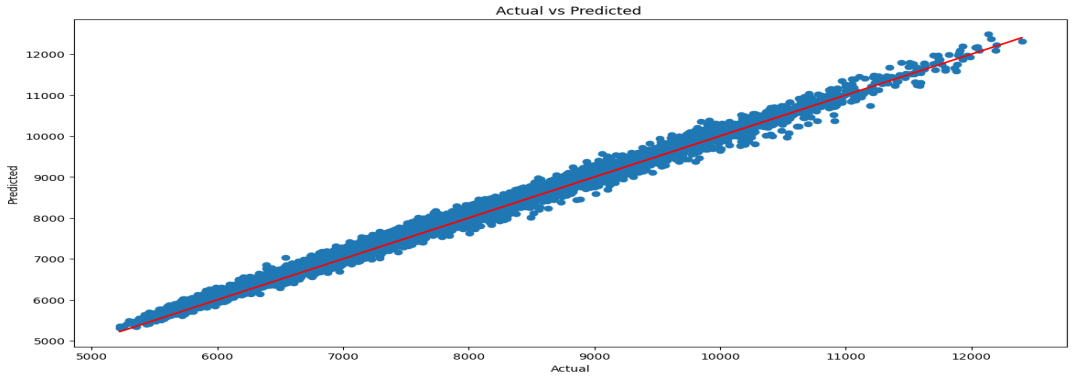

Figure 5.7: Actual vs Predicted demand using Prophet model

In the Figure 5.7, the actual and the predicted demand is shown. And we can see that the model has done a good job in predicting the demand for the test data as the actual and predicted demand are very closely concentrated around the line.

5.2.7 Model Forecast

The Fig 5.8 shows the forecast for 17 days for March 2021:

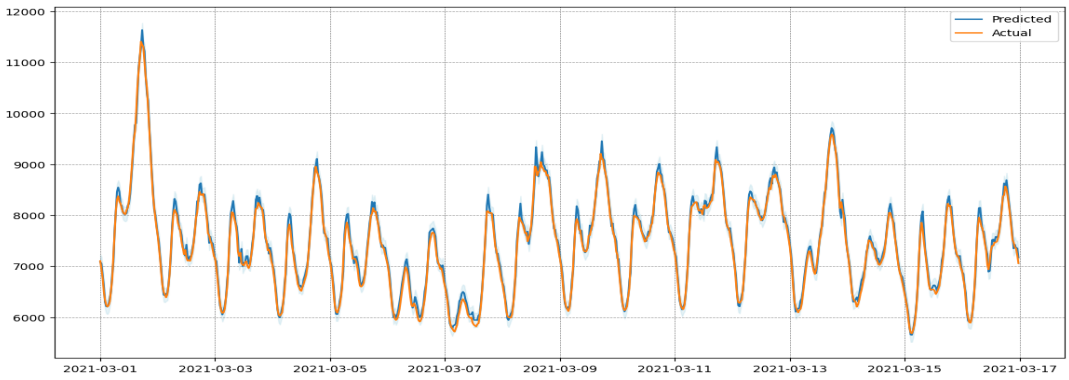

Figure 5.8: Forecast demand using Prophet model

Chapter 6

Model comparison and Discussion

XGBoost excels in handling complex non-linear relationships but does not account for time series components like trends and seasonality. It requires manual feature engineering to forecast time series data, such as transforming hours using Cosine and adding a demand lag column to capture temporal dependencies. However, Facebook Prophet is designed with time series forecasting in mind. Prophet deploys additive models that naturally incorporate trends (yearly, monthly, and daily). It is powerful when handling time series.

Table 6.1: XGBoost, Prophet and AEMO Model Comparison

Model	**RMSE**	**MAE**	**MAPE**
XGBoost	88.13	67.04	1.0%
Prophet	153.13	132.16	1.8%
AEMO	173.61	131.69	2.0%

XGBoost and Facebook Prophet had lower errors compared to the AEMO forecast model. Since both RMSE and MAE are lower for XGBoost and Facebook Prophet, Both models generally perform better across small and large errors. The plot below shows how the models predicted actual demand. We can see that XGBoost follows the actual demand more closely.

Both models are flexible and can process any time interval. XGBoost's performance is based on the feature engineering approach. The features must capture the dynamics of electricity demand at this granularity for short intervals. Facebook Prophet automatically detects and captures patterns in half-hour electricity demand data. Facebook Prophet is more user-friendly than XGBoost. Its components are understandable, making understanding the factors influencing the forecast easier. While XGBoost requires careful feature engineering and parameter tuning, model interpretation is challenging due to the ensemble nature of the model.

Facebook Prophet is tailored for time series data. It has built-in mechanisms to capture trends, making it a robust model for electricity demand forecasting at different intervals. XGBoost is highly accurate in many domains and is one of the most used models in machine learning competitions. However, without adequate feature engineering to address temporal relationships, XGBoost may not perform as effectively as models designed for time series.

The XGBoost and Facebook Prophet models provided better forecasts than the model used by AEMO. At first glance, both XGBoost and Facebook Prophet are

outright better models. However, Several factors need to be considered before making such a claim. Firstly, AEMO's model might be more general-purpose and work across various regions, time frames, and applications. The two models we built are tailored to a specific dataset, time frame, and region, allowing the models to focus on the unique characteristics of electricity forecast in that context. This specialisation might explain the better performance in this case.

Hyperparameter tuning is affected by the conditions and adapts to the exact dataset and timeframe. A small change in a hyperparameter, like the learning rate or tree depth, can significantly impact results. The models we built may be finely tuned to capture the patterns in electricity demand in this context. However, the AEMO model could be built and tuned to handle various scenarios and time frames.

The models we built focus on short intervals. AEMO model might be optimised for longer-demand patterns due to operational needs. We intended to tune for forecasting short-term demand, while the AEMO model could be tuned for long-term stability. This could explain why our model performed better when forecasting a short-term interval.

Our models are optimised for a specific time window (30 minutes). Thus, they are suited to detecting patterns in that period. A more general model might need help with extreme cases like sharp demand spikes or sudden peak-hour drops. This could explain why our model performed better, but this comes at the cost of model forecast period flexibility. Our model will struggle if used to forecast a long-time horizon compared to the AEMO model.

Figures 6.1 and 6.2 compare the forecasted demand among the AEMO, XGBoost, and Prophet models for a single day and for an extended period for seventeen days respectively. The XGBoost model follows the actual demand more closely than the Prophet and AEMO models.

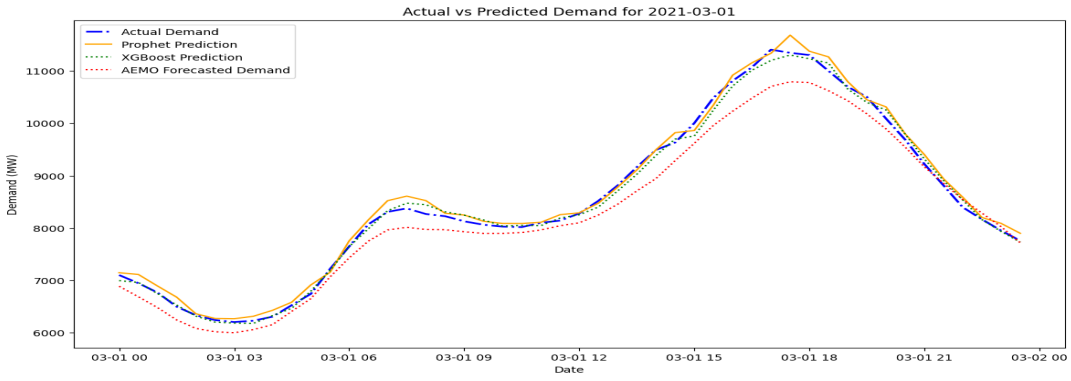

Figure 6.1: Forecast demand comparison among AEMO, XGBoost and Prophet

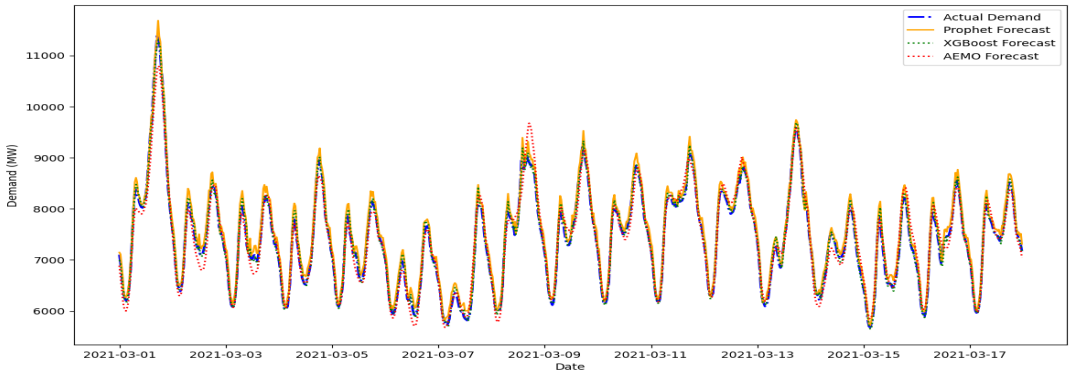

Figure 6.2: Forecast demand comparison among AEMO, XGBoost and Prophet

The XGBoost model was able to predict the peaks and valleys, which are often more volatile and challenging to predict. As we observe from the Figure 6.1, during the peak hours from 9am to 3pm both of our models forecasted better but during the off-peak hour XGBoost model forecasted better than Prophet model Also we observe the AEMO model is underpredicting the demand.

Chapter 7

Conclusion and Future Work

We built, tested, and compared popular machine-learning models – XGBoost and with an additive model – Prophet. XGBoost is highly flexible and known for its high accuracy in many domains, while Facebook Prophet is a time-series-specific model that excels in capturing trends. The goal is to see how these different models forecast short-term electricity demand.

We found that both models performed well with very close results. Facebook Prophet is easier to use and more efficient when forecasting electricity demand. It offers a more user-friendly approach because it's built to handle time series data with clear seasonal patterns. However, XGBoost can deliver competitive results at the cost of substantial feature engineering.

We also compared the two models with the AEMO forecast to benchmark our performance against industry standards, and our models performed better. However, we do not suggest that our models are superior to AEMO. There are many valid time horizons for electricity demand forecasting where the current AEMO model will outperform the models we build. We trained our model on a specific dataset, while the AEMO model is trained on more extensive and complex data. The AEMO model is also built to handle various scenarios. However, our model is built for one specific scenario.

The results highlight how tailoring a model to a specific forecasting problem can improve accuracy. By creating particular models, we can better address electricity demand's short-term, high-frequency nature. This report emphasises the importance of customisation in forecasting models. Comparing multi-use multi-timeframe models and single-use case models is work that needs to be done in the future, along with incorporating the impact of photovoltaic systems and population and economic growth.

For future work, we recommend expanding the dataset to include more features like weather data, economic indicators, and population growth. This will help the model capture more complex patterns and improve forecasting accuracy. We also want to recommend the cost-benefit analysis of implementing the models in real-world scenarios. This will help stakeholders to choose the best model for their specific needs. We will also explore popular deep learning models like LSTM and other pre-trained models like DeepAR for short-term electricity load forecasting.

References

References

[Adapt NSW(2024)] Adapt NSW, 2024. NSW — AdaptNSW – climatechange.environment.nsw.gov.au. https://www.climatechange.environment.nsw.gov.au/my-region/nsw. [Accessed 21-09-2024].

[AEMO(2022)] AEMO, 2022. Forecasting approach – electricity demand forecasting methodology. https://aemo.com.au/-/media/files/electricity/nem/planning_and_forecasting/nem_esoo/2023/forecasting-approach_electricity-demand-forecasting-methodology_final.pdf. [Accessed 21-09-2024].

[AEMO(2023)] AEMO, 2023. Forecasting Approach: Electricity Demand Forecasting Methodology. Technical Report. Australian Energy Market Operator. URL: https://aemo.com.au/-/media/files/electricity/nem/planning_and_forecasting/nem_esoo/2023/forecasting-approach_electricity-demand-forecasting-methodology_final.pdf. accessed: 2024-09-22.

[AEMO(2024)] AEMO, 2024. Forecasting and planning guidelines. URL: https://aemo.com.au/en/energy-systems/electricity/national-electricity-market-nem/nem-forecasting-and-planning/forecasting-approach/forecasting-and-planning-guidelines. accessed: 2024-09-30.

[Ahmad et al.(2022)Ahmad, Ghadi, Adnan and Ali] Ahmad, N., Ghadi, Y., Adnan, M., Ali, M., 2022. Load forecasting techniques for power system: Research challenges and survey. IEEE Access 10, 71054–71090. doi:10.1109/ACCESS.2022.3187839.

[Ambika(2023)] Ambika, 2023. Xgboost algorithm in machine learning. URL: https://medium.com/@ambika199820/xgboost-algorithm-in-machine-learning-2391edb101ce.

[avanwyk(2022)] avanwyk, 2022. Encoding cyclical features for deep learning. URL: https://www.kaggle.com/code/avanwyk/encoding-cyclical-features-for-deep-learning.

[Blue(2024)] Blue, C., 2024. Peak and off-peak electricity times. https://www.canstarblue.com.au/electricity/peak-off-peak-electricity-times/. Accessed: 2024-09-30.

[Brownlee(2017)] Brownlee, J., 2017. How to decompose time series data into trend and seasonality. URL: https://machinelearningmastery.com/decompose-time-series-data-trend-seasonality/.

[Brownlee(2020a)] Brownlee, J., 2020a. A gentle introduction to the gradient boosting algorithm for machine learning. URL: https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning. accessed: 2024-10-01.

[Brownlee(2020b)] Brownlee, J., 2020b. How to use xgboost for time series forecasting. URL: https://machinelearningmastery.com/xgboost-for-time-series-forecasting/.

[Chen et al.(2015)Chen, Wang, Liu, Wang and Liu] Chen, C., Wang, P., Liu, H., Wang, Y., Liu, Z., 2015. Electricity load forecasting: A review. Renewable and Sustainable Energy Reviews 41, 515–527.

[Chen and Guestrin(2016)] Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system. URL: https://arxiv.org/pdf/1603.02754.

[Department of Climate Change and Water(2023)] Department of Climate Change, Energy, t.E., Water, 2023. Australian energy statistics: Renewables. URL: https://www.energy.gov.au/energy-data/australian-energy-statistics/renewables. accessed: 2024-09-30.

[Esri(2016)] Esri, 2016. How xgboost algorithm works—arcgis pro — documentation. URL: https://pro.arcgis.com/en/pro-app/latest/tool-reference/geoai/how-xgboost-works.htm.

[Fan and Hyndman(2012)] Fan, S., Hyndman, R., 2012. Short-term load forecasting based on a semi-parametric additive model. IEEE Transactions on Power Systems 27, 134–141. doi:https://doi.org/10.1109/tpwrs.2011.2162082.

[McCulloch et al.(2001)McCulloch, Tsay and Wu] McCulloch, R.E., Tsay, R.S., Wu, P., 2001. Forecasting electricity demand: A comparative study. International Journal of Forecasting 17, 3–18.

[NIST(2020)] NIST, 2020. 1.3.5.12. autocorrelation. URL: https://www.itl.nist.gov/div898/handbook/eda/section3/eda35c.htm.

[NSW EPA(2021)] NSW EPA, 2021. Energy consumption 2021. URL: https://www.soe.epa.nsw.gov.au/all-themes/human-settlement/energy-consumption-2021. accessed: 2024-09-22.

[Penn State University(2024)] Penn State University, 2024. Lesson 10: Interaction terms. URL: https://online.stat.psu.edu/stat462/node/188/. accessed: 2024-10-01.

[Rafferty(2023)] Rafferty, G., 2023. Forecasting Time Series Data with Prophet. Packt Publishing Ltd. URL: https://www.packtpub.com/product/forecasting-time-series-data-with-prophet/9781803234675.

[Readthedocs(2022)] Readthedocs, 2022. Xgboost parameters — xgboost 2.1.1 documentation. URL: https://xgboost.readthedocs.io/en/stable/parameter.html.

[scikit-learn(2024)] scikit-learn, 2024. TimeSeriesSplit. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html. accessed: 2024-10-01.

[Suo et al.(2019)Suo, Song, Dou and Cui] Suo, G., Song, L., Dou, Y., Cui, Z., 2019. Multi-dimensional short-term load forecasting based on xgboost and fireworks algorithm, in: 2019 18th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), IEEE Computer Society, Los Alamitos, CA, USA. pp. 245–248.

[Taylor and McSharry(2009)] Taylor, J.W., McSharry, P.E., 2009. Forecasting electricity demand: A review of methods and approaches. Energy Policy 37, 1279–1292.

[Taylor and Letham(2017)] Taylor, S.J., Letham, B., 2017. Facebook prophet: Forecasting time series with additive models. Journal of the American Statistical Association .

[UNSW-ZZSC9020(2024)] UNSW-ZZSC9020, 2024. Project data. https://github.com/UNSW-ZZSC9020/project/tree/main/data. Accessed: 2024-09-30.

[Wang et al.(2016)Wang, Wang, Xu, Zhang, Liu and Wang] Wang, Y., Wang, L., Xu, L., Zhang, B., Liu, X., Wang, J., 2016. A review of electricity load forecasting techniques. Renewable and Sustainable Energy Reviews 56, 176–189.

[Wrigley(2019)] Wrigley, K., 2019. Peak and off-peak electricity times - tariffs and rates. URL: https://www.canstarblue.com.au/electricity/peak-off-peak-electricity-times/.

Appendix

Figures

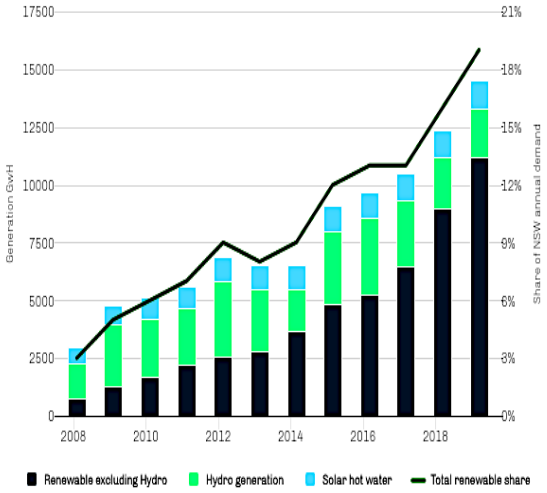

Figure 7.1: Renewable fuel sources [Source: Derived from Department of the Environment and Energy, Australian Energy Statistics, Table O, June 2021]

Tables