Prague University of Economics and Business

Faculty of Informatics and Statistics

# PREDICTIVE MODELLING ELECTRICITY PRICES FOR SHORT-TERM AND LONG-TERM HORIZONS, THE CASE OF THE CZECH REPUBLIC

Study programme: Masters of Economic Data Analysis

Specialization: Data Analysis and Modelling

Author: Christian Billinton

Supervisor: Ing. Karel Helman, Ph.D.

Prague, June 2024

## Acknowledgement

I would like to share my gratitude for my supervisor, Ing. Karel Helman, Ph.D. for his support and judgement not just through writing this thesis, but throughout my studies.

I would like to give thanks to the circle of support I've been fortunate to have throughout my studies. My parents for their undeniable care and wisdom, my Uncle Mike for always being there for me as a role model and my partner Ivana for her unwavering love, that I could not do without.

## Abstract

Accurate forecasts are a crucial aspect of many industries such as electricity and energy markets for which this paper studies. While many markets have gone through turbulent time, this study aims to apply Autoregressive Integrate Moving Average (ARIMA), Support Vector Regression (SVR) and Long-Short Term Memory (LSTM) models to predicting the daily wholesale electricity prices from the Czech day ahead market. This thesis employs the use of a rolling interval approach to predicting across the year 2022 for short-term horizons and long-term horizon, in which the short-term horizon is defined as the next day's price, while the long-term horizon being the forecast for the multi-step of the next month. A variance reduction method of log transforming the data is used, while models are also used with the un-transformed data. As a result, Support Vector Regression Model outperformed the other models across both time horizons, while the ARIMA and LSTM model provided closely comparable results on average across the year.

## Keywords

Electricity Prices Forecasting, Time Series, ARIMA, Support Vector Regression, Long-Short Term Memory Models, Rolling Interval

## Table of Contents

## List of Figures

## List of Tables

## List of Abbreviations

| | |
|---|---|
| ARIMA | Autoregressive Integrate Moving Average |
| AR | Autoregressive |
| ACF | Autocorrelation Function |
| ADF | Augmented Dickey Fuller |
| AIC | Akaike Information Criterion |
| BIC | Bayesian Information Criteria |
| I | Integrated |
| KPSS | Kwiatkowski-Phillips-Schmidt-Shin |
| LSTM | Long-Short Term Memory |
| MA | Moving Average |
| MAE | Mean Absolute Error |
| OTE | Operátor Trhu s Elektřinou |
| PACF | Partial Autocorrelation Function |
| RMSE | Root Mean Squared Error |
| SVM | Support Vector Machine |
| sMAPE | scaled Mean Absolute Percentage Error |

## List of Symbols

| | |
|---|---|
| $A_t$ | Activation function |
| $B$ | Backshift operator |
| $b$ | Bias |
| $C$ | Cost for SVM |
| $\tilde{C}_t$ | Candidate memory cells |
| $d_t$ | Deterministic trend |
| $h_t$ | Output in LSTM |
| $k$ | Number of parameters |
| $r_t$ | Random walk |
| $w$ | Weights |
| $w_{kj}$ | Weight matrix |
| $X_t$ | Input vector for LSTM at time *t* |
| $y_t$ | Observed value of time series in time t |
| $Y_t$ | Theoretical random variable at time *t* |
| $\hat{y}_t$ | Predicted value at time *t* |
| $\alpha_i$ | Sample point within SVM tube |
| $\varepsilon$ | Error width in SVM |
| $\xi_i$ | Slack variables in SVM |
| $\phi_p$ | AR coefficient of order *p* |
| $\psi_q$ | MA coefficient of order ps |

# 1. Introduction

In today's age, time series predictive modelling has taken on the rapid acceleration of interest in the business world and research. Energy markets have taken the stage as the central focus of the past decade due to the readily available data and the liberalization of markets. With the rise of war in Ukraine and extreme weather events (Matalucci, 2021), the need for a way through uncertainty comes with the need for predictive modelling of electricity prices, as many actors have a stake in consistent pricing based on their needs.

There are many reasons why electricity markets and price predictions are an essential area of study, as our daily lives depend on mobile devices and electronic systems. Major disruptions due to climate events have proven to impact grids and create shocks in electricity prices. These effects create insecurity in the supply field (Keefe, 2022) and pose potential price prediction challenges.

The impact of technological advancements has been another reason for the study of electricity prices, as the digitization of our lives involves building a reliable power supply through exceeding demand. We have seen this in the cases of the rapid adoption of electric vehicles (EVs) and even in the case of Bitcoin mining. The crypto craze has produced economic spillovers in retail consumer energy prices (Benetton, Compiani and Morse, 2023). The automotive industry has also become a relevant factor in energy grids as the rise of EVs has been a prime example of the electrification of our lives, and a significant amount of research to extensive adoption. There are costs to such adoption, such as charging patterns for EVs that happen outside of peak hours and create surmounting demand at night, yet innovative approaches exist to solve these problems (Needell, Wei and Trancik, 2023).

With different economic actors fulfilling costly and complex operations to satisfy energy markets, the walk along the precipice between economics and engineering occurs (Stoft, 2011). This pivotal commodity for society involves complex networks of actors participating at scale in generation, distribution, and demand of electricity. This comes with several different needs, such as the pricing of electricity and the time horizons required. There are also external implications that changes in electricity prices can bring about, such as in firm-related performance (Gazzani and Ferriani, 2022). This blending of fields has also created a large number of applications for time series modelling, beyond prices such as for reliability (Billinton, Chen and Ghajar, 1996).

This thesis aims to define and compare the effectiveness of two modelling techniques popularized for different needs at different periods of electricity prices using spot price data for the Czech Republic. The time series used covers five years, from 2018 to the end of 2022. Due to the volatility of energy markets, actors require accurate price predictions for short-term and long-term horizons. Because of these needs, this thesis will focus on these two horizons for univariate modelling defining: long-term periodicity as one month in advance and the short-term horizon for next-day values. The literature for electricity price forecasting has seen an explosion of deep learning models of Artificial Neural Networks, Recurrent Neural Networks, and LSTM models, but further use of alternative models provides ample room for exploration. The analysis in this thesis will aim at comparing univariate time series predictive accuracy across three modelling techniques of ARIMA, Support Vector Machines, and Long-Short Term Memory Networks for the *better* model in terms of accuracy, as defining the best model will never have a concrete consensus in its subtle beauty. The data will be compared in using the common variance reduction technique of log transforming benchmarked against un-transformed data across a rolling interval. The software used in this study is the R programming language (R Core Team, 2024).

## 1.1 Introduction to Electricity Pricing

Consuming electricity in our day-to-day lives is as simple as pressing a button on the kettle, but how does this electrical power and the prices that end up in our utility bill in complex systems for our needs? Firstly, energy is measured in Watthour (Wh) and priced in Megawatt hour (MWh), which is one thousand Wh. The electricity we increasingly interact with is known as a secondary source (U.S. Energy Information Administration, 2023), meaning it is produced by converting a primary energy source into electrical power, such as coal, natural gas, nuclear, wind, and solar.

At the most foundational level, the many steps for electrical power to arrive at our fingertips require all the acts of power generation, transmission, and distribution (Carbaugh and Sipic, 2017). Generators of electricity use various high-energy sources. This is the highest-cost element, with the type of power plants used being categorized into two respective types: baseload generators and peaking power plants. The role of the baseload generators is to produce at a constant rate as large pants with a high fixed cost and lower marginal cost. The role of the baseload powerplant is to maintain the minimal demand over time.

On the other hand, peaking power plants tend to be called into action during peak demand, carrying low fixed costs and high marginal costs. These systems are meant to ensure the balance of supply and demand in the market to be as efficient as possible. Electricity storage is non-existent, meaning what is produced and traded must be consumed, leaving little room for waste.

Through this system, electricity is bought and sold on wholesale markets. Wholesale markets start with the generator that will produce and sell to resellers, which are, in most cases, utility companies that connect and deliver to meet end users' demands. The lack of storage means that market participants participate heavily in trading over different time horizons and in futures, day ahead and intraday markets. It also means that systems operators must balance what is demanded with what is sold through a bidding structure. This European Union is unique in that it has the most extensive electrical grid in the world, following the founding principle of free access to goods through its grid via the entity ENTSO-E.

A fascinating occurrence in electricity markets are periods of negative prices occurring due to economic market failures, where producers constrained by inability to store electricity need to make payment to a buyer (Kaňková and Koľveková, 2020). The introduction of the negative prices on the day ahead market by the Czech market operator OTE occurred in cooperations with the Slovak market in 2012 in order to accept negative and zero price bids. This phenomenon occurs when an imbalance in supply occurs due to greater amounts of electricity is generated than is demanded in the market. This creates an incentive for producers to correctly adjust generation and for buyers to consume electricity to ensure grid stability.

Futures contracts are agreements to buy or sell a commodity on a future day with a price and amount that is fixed in the agreement. This can be up to six years ahead for electricity markets, providing a means for risk management over long-time horizons. On the other hand, the spot market comprises day-ahead prices for trades focused on the following day's intraday prices, which are for trades on the same day, creating the short-term horizon. Electricity price forecasting rise in popularity precisely stems from the need to “Know when to hold them and when to fold them,” in the words of the singer Kenny Rogers.

A key differentiation of the market for electricity prices from other commodities or financial markets is based on the structure of the market in which systems operators needing advanced notice for scheduling orders to be made before a specific deadline on the day-ahead market (Weron, 2014). This also means that the hourly data is not a time series whereby the value at hour t is not proceeded by the hour *t*, making it a 24-hour panel of cross-sectional daily data (Huisman, Huurman and Mahieu, 2007).

## 1.1. Description of the Data and Why

This thesis uses data from the Czech Republic's energy market for data selection, with a highly coal-dependent energy sector, accounting for 46%. This is a primary nationally available resource and provides security over supply not achieved with natural gas. Renewables play a minimal role in the energy mix, accounting for roughly 16%, while the central focus is on the increased share of nuclear and natural gas to phase out coal (OECD, 2021). The Czech Republic also ranks as a significant energy exporter of roughly 26 billion KW in 2021, making it the sixth biggest exporter in the world (U.S. Energy Information Administration, no date).

The Czech Republic wholesale market is fully liberalized with complete separation between generation and transmission, yet the largest energy company in the Czech Republic is CEZ, which holds a major share of the market. The wholesale market has seen significant growth in the number of participants (OECD, 2021), reaching 397 in 2019. As a result, the current total traded volume accounts for 81% of the spot market.

While options markets play a role in the buying and selling electricity, the day ahead market plays a crucial role in the analysis of prices. For this reason, the analysis of this thesis focuses on using daily averaged day-ahead market price data for predictive modelling over five years, from 2018 to the end of 2022. This is provided by the Czech electricity and gas market operator (OTE), the market operator for the Czech Republic, ensuring the data processing and exchange of the spot market for both electricity and gas markets run smoothly when connecting generators to resellers. Using two forecasting horizons, the analysis will use two different rolling interval approaches to predict the respective lengths.

# 2 Electricity Price Forecasting Methodology

Much of the literature surrounding electricity price forecasting centres on a breakdown of models into statistical models and machine-learning models. Yet, as a unique financial commodity, it rests in the same realm as financial data yet bleeds into areas of scientific engineering. As an industry, the forecasting of energy-related data has seen the rise of competitions hosted by the Institute of Electrical and Electronic Engineering (IEEE) for hybrid energy of the European Statistics Awards for nowcasting. Electricity price forecasting, though, originated in statistical modelling, with some of the first models employing classical regression models, which, with the advancement of computational power, a greater degree of use machine are employed and covered in researching electricity prices (Weron, 2014; Jędrzejewski *et al.*, 2022).

This literature review will follow two modelling techniques employed in this paper's analysis from two groupings of Statistical and Computational Intelligence (CI) approaches. The statistical approach used to generalize approaches is the most commonly used to forecast day ahead prices using mathematical combinations of the previous day price. This study explores the popular ARIMA approach from this grouping, based on past observations, to make predictions of the future. Support Vector Regression models and Long Short-Term Memory Neural Networks fall into the grouping of Computational Intelligence approaches. Used synonymously with terms such as Artificial intelligence or Machine Learning in the literature, this study will solely use CI and Machine Learning as terms for this grouping of models throughout the paper. While commonly used for classification problems, these approaches provide a comparable approach to forecasting electricity prices as they can take univariate inputs and provide accurate time series forecasts.

Alternatively, other groupings made for forecasting electricity prices not covered within this study take on approaches outside of using a univariate series of the price to generate a forecast. These approaches are described as Multiagent, Fundamental, and Reduced Form models, which are briefly described in the following section.

## 2.1 Time Series Forecasting

Time series analysis is the study of observations arranged in chronological order, which can be thought to be a realization of a stochastic process of random variables $Y_t$ ,as a result of the underlying data-generating process (Kirchgässner, Wolters and Hassler, 2012). A discrete time series is generated over a distinct interval between time periods, while a continuous time series contains irregular space intervals between observations.

The intuitive part of the time series that generates practical interest in its analysis is that adjacent observations are dependent across time (Box, Jenkins and Reinsel, 2008). This allows for great potential in analyzing and forecasting univariate time series based on learnings from the past.

When predictive performance is being evaluated, an important distinction needs to be made when it comes to the use of in-sample and out-sample measures are used. We can discern the in-sample period as the data that is used for the initial estimation and model selection, while the out-sample period is a period of observations held out to be used to evaluate the performance of the prediction. In-sample measures for predictability

### 2.1.1 Predictive-Accuracy Measures

Predictive accuracy aims at measuring the loss that a model would have in predicting unseen data (Cerqueira, Torgo and Mozetič, 2020). The aim is to quantify the accuracy of the model for comparative interpretability with the out-sample observations.

Accuracy measures can be grouped into several different groups (Hyndman, 2006). The most commonly used are scale-dependent, which utilize the same scale as the data. Percentage errors on the other hand have the advantage of being independent of scale making for easier interpretability across different time different data.

**Symmetric Mean Absolute Percentage Error (sMAPE)**

A common percentage error accuracy measure is the Mean Absolute Percentage Error which is similar to the scale dependent measure Mean Absolute Error (MAE) which takes the mean of the absolute error. But it takes the percentage error rather than the error between the predicted and observed values. This method has been noted (Hyndman, 2006) not to handle near-zero values well and penalize negative values more than positive values. Electricity prices experience negative values in special situations making this an unsuitable method for electricity price forecasting. It is more common to use a symmetric MAPE in its place. This is calculated through (1)

$$sMAPE = \frac{100}{n}\sum_{t=1}^{n}\frac{|\hat{y}_t - y_t|}{\frac{|y_t| + |\hat{y}_t|}{2}} \tag{1}$$

Where $\hat{y}_t$ is the predicted value at time $t$ and the $y_t$ is the actual value at time $t$

**Root Mean Squared Error (RMSE)**

Another frequently used scale-dependent accuracy measure is the root mean squared error. Its measurement is useful as it is in the same units as the data, yet can be sensitive to outliers of errors.

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\widehat{y_\iota} - y_t)^2} \qquad (2)$$

Where $\hat{y}_t$ is the predicted value at time $t$ and $y_t$ is the actual values at time $t$

### 2.1.2 Rolling Interval Prediction

A rolling interval prediction is an evaluation technique where the forecasting origin is updated consecutively and moved ahead along the time series. Several different approaches to this can be made (Tashman, 2000). The rolling interval method may be adjusted for several train and test samples. While forecasting can take on subjective reasoning for the methods applied, the use of a rolling interval when forecasting helps ensure that data can be used for training and testing models in a way that the models may be employed realistically to train on a specified interval and predict the next day.

There are many terms for this approach, such as rolling interval, rolling origin and time series cross-validation. For consistency, this study will stick with using the term rolling interval throughout the course of the paper.

A forecaster may hold the out-sample sample of constant size and increase the length of the in-sample, ensuring constant length of the forecasted horizon while capturing the estimated. The importance of performing a rolling interval prediction lies in the susceptibility for corruption of a single point forecast due to the unique developments in the single train window. The rolling interval provides a measure to ensure that the forecast is not influenced by a single cycle.

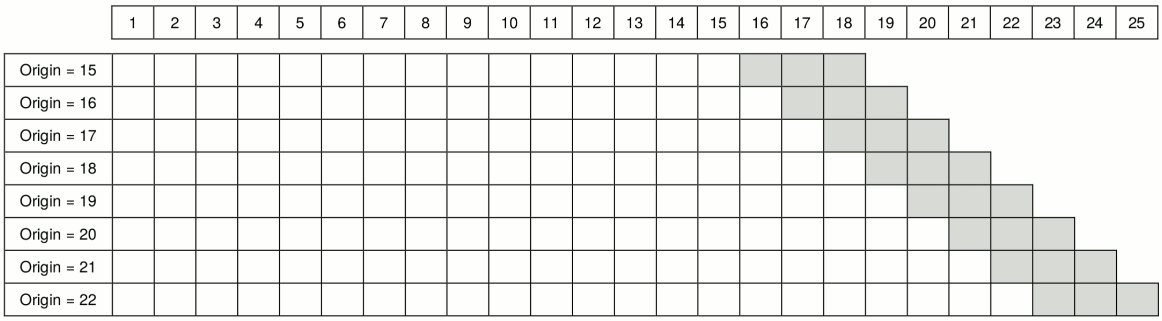


*Figure 1: Rolling Interval with Constant Out-Sample Size (Svetunkov and Petropoulos, 2017)*

Another method of implementing a rolling interval can be having a non-constant hold out sample that can be useful in the case of smaller sample sizes, as seen in figure 2.

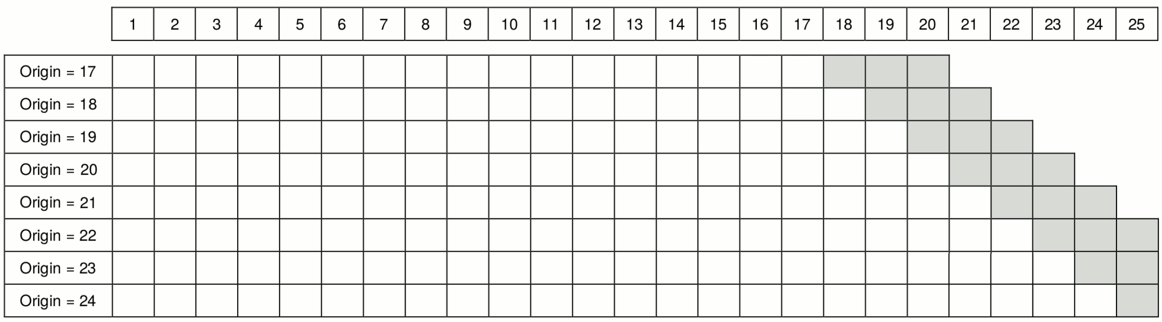


*Figure 2: Rolling Interval with Non-Constant Out-Sample Size (Svetunkov and Petropoulos, 2017)*

The previous two methods have a non-constant sample size as it grows across each interval. As seen in Figure 3, we can set the in-sample and the out-sample as fixed to roll over, keeping fixed lengths while incorporating the next sample value. The constant in sample size removes the observation of the length of the data, cleaning out data from a separate period.

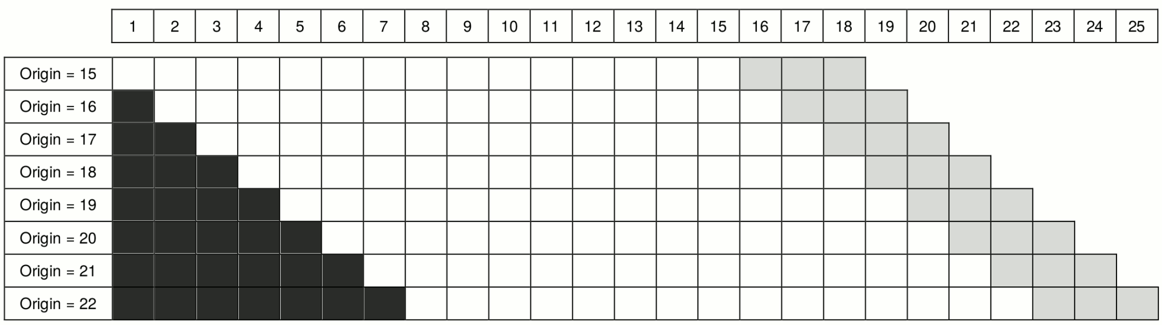


*Figure 3: Rolling Interval with Constant In-sample and Out-sample Size (Svetunkov and Petropoulos, 2017)*

The selection of the size of the in-sample and out-sample is generally a subjective choice up to the forecaster. This subjectiveness is diminished in the use of rolling interval on the forecasted accuracy (Fildes and Makridakis, 1995).

Rolling interval methods can use different methods of conducting the forecast, such as an initial estimation or the use of re-estimation for each window. Initial estimation utilizes the model estimated across the first interval to forecast each proceeding interval. This approach closely resembles a multi-step direct prediction method described in the chapter 2.1.3. Re-estimation for each interval on the other hand, allows for errors based on unique developments of the fitted interval. It is preferable to recalibrate on each window of the model, yet the drawback is the difficulties of re-estimation for each interval make modelling over long time intervals computationally demanding.

### 2.1.3 Step Prediction

The number of steps within a prediction determines the future time-period we are attempting to predict. We can split the strategy of defining a step for our forecast into two groups of a single-step prediction, which is the most common focus of many electricity price forecast studies. The second is the use of multi-step prediction strategies as different methods are employed (Taieb *et al.*, 2011). The multi-step prediction attempts to predict $H$ periods into the future, creating a longer prediction. The three main strategies for multi-step forecasting are Recursive, Direct and Multi-Input Multi-Output Strategies.

**Single Step Prediction**

The single-step prediction forecasts a single period right after the sample length ends. This is relatively straightforward and, for practical purposes, can be understood in the case of daily time series as a prediction of the next day's value, or a single value predicted for $y_{t+1}$. Due to the need for a constant out-sample size of 1 when using single step predictions the method seen in Figure 2 of a non-constant out-sample is not appropriate as the steps would simply shift by a single observation.

**Multi-Step Recursive Prediction**

The multi-step recursive strategy, similar to the rolling interval, involves the use of conducting a single-step prediction and then applying the forecasted value as input to forecasting the next step. The problem is that the error of the predicted values is then included as an input to the future predictions of the model, decreasing accuracy as the forecasting horizon increases.

**Multi-Step Direct Prediction**

The multi-step direct strategy applies a single prediction for $n$ independent steps from individual trained models for that specific step. The direct approach does not use approximated values unlike the recursive strategy, which induces conditional independencies among each forecasted series value. As models are trained for individual forecasted step resulting in $N$ models for $n$ time steps, this method can experience increasing computational complexity.

**Multi-Step Multi-Input Multi-Output Prediction**

The Multi-Input Multi-Output (MIMO), unlike the single-output approaches previously mentioned, focuses on a single model which is trained to predict multiple steps simultaneously in its output. This strategy avoids the independence between each predicted value with the Direct approach and the bias of errors going towards predicted values through the Recursive approach. Yet, the downside of this approach is that each step is constrained to a single model, reducing the flexibility of the approach across all time steps.

## 2.2 ARIMA Models

One of the most prominent modelling techniques used for modelling univariate time series. Originally formulated by George E.P Box and Gwilym M. Jenkins (Box *et al.*, 2015), often referred to as Box-Jenkins models, are wildly hailed as one of the most popular times series models, which are often used as a benchmark of comparison with alternative methods (Siami-Namini, Tavakoli and Siami Namin, 2018). ARIMA models are an expansion of ARMA models made up of a combination of Autoregressive (AR), Integrated (I) and Moving Average (MA) process.

In the literature of electricity price forecasting, these models fall into the classification of Statistical models (Weron, 2014), which also encompasses common practices employing exponential smoothing and regression models. Their effectiveness lies in using statistical models for electricity prices that are reliant on learning from past observations to forecast the future. Electricity prices experience strong seasonality through spiky periods, which brings its disadvantages as statistical models tend to experience difficulties with high levels of variability, as in the case of electricity prices.

The notation used to express ARIMA models commonly is in the form of backshift notation(Hyndman and Athanasopoulos, 2018), for cleaner interpretation. The key element of the backshift notation being the backshift operator $B$ which can be understood as a coefficient shifting the observation at time $t$ to its previous observation demonstrated in the following equation.

$$BY_t = y_{t-1} \tag{3}$$

### 2.2.1 Autoregressive Model (AR)

The autoregressive process is that of the linear combination of the past values of our univariate variable. The model of order $p$ can be expressed where $\phi_p$ is a finite set of parameters.

$$y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \varepsilon_t \tag{4}$$

Where $\varepsilon_t$ is the white noise error term. Its formulation can also be made in its backshift notation which can be formulated as:

$$\phi(B)Y_t = \varepsilon_t \tag{5}$$

Based on the formulation, higher a degree of $p$ increases the number of previous observations used for estimation of the current term and increase the complexity of the model.

### 2.2.2 Moving Average Model (MA)

The moving average process can be expressed through the order $q$ and weights $\psi$, which can express a moving average of the past forecasting errors in the model.

$$y_t = c + \varepsilon_t + \psi_1 \varepsilon_{t-1} + \psi_2 \varepsilon_{t-2} + \cdots + \psi_q \varepsilon_{t-q} \tag{6}$$

Which can be formulated in its backshift notation as:

$$y_t = \theta(B)\varepsilon_t \tag{7}$$

### 2.2.3 Autoregressive Integrated Moving Average Model

The *ARMA* model is a combination of the autoregressive model and the moving average model which provides the advantage of greater flexibility in fitting the model of stationary time series. For non-stationary time series the incorporation of the order of integration (*I*) is included to formulate an ARIMA model. The reason for using the word integrated or the non-seasonal difference of the data is to handle non-stationarity which many series of economic data exhibit. The order of the terms is applied in the respective order of $AR$, $I$, and $MA$ components under the notation:

$$ARIMA(p, d, q) \tag{8}$$

The full model can be written as:

$$y'_t = c + \phi_1 Y'_{t-1} + \cdots + \phi_p Y'_{t-p} + \psi_1 \varepsilon_{t-1} + \cdots + \psi_q \varepsilon_{t-q} + \varepsilon_q \tag{9}$$

Where $Y'_t$ is the differenced series which can be formulated as $y' = (1 - B)^d y_t = \nabla^d y_t$ where the difference operator is represented as $\nabla = 1 - B$. This powerful model for describing nonstationary and stationary processes can be formulated as backshift notation with the inclusion of the integrated process for non-stationary differencing

$$\phi(B)\nabla^d y_t = \theta(B)\varepsilon_t \tag{10}$$

Thus, the ARIMA model can be seen as the summation of the individual components of AR and MA.

### 2.2.4 Model Identification

Model identification for an ARIMA model is commonly done across numerous statistical measures to provide a general order of the model's terms. For AR and MA orders, this process can be the terms, and it can be done through subjective selection-based identification of the autocorrelation plots or through automated processes of selecting a best-fitting model. The order of differencing may be determined based on stationarity and unit root testing. In the end, the most crucial aspect concerning the method for model selection lies with the end result.

A procedure to selection of a suitable predictive model can be made through the series of steps (Contreras *et al.*, 2003) taken to estimate the model presented in the previous sections. These steps provide general guidance to finding a suitable model, yet the accuracy results of the prediction are the most important aspect of finding the most suitable model. These steps for ARIMA modelling are:

1. **A model is identified for the observed data.**

The formulation of the order of the model can be identified through testing and certain procedures. Transformations that are applied are identified in this step to reduce variance. The common practice being to take a log transformation, or rates of dynamics commonly used in financial markets, such as returns. Reducing the variance is key in electricity markets where large spikes in the price commonly occur, yet unlike traditional financial data, electricity prices on occasion dip below zero, which requires some alternative procedures such as in shifting through a constant.

Identifying the order of ARIMA process can be done through the use of autocorrelation plots and stationarity testing. The order of the models AR and MA terms may be estimated from the results of the Correlogram dealing with the autocorrelation of lagged values. This can also be used to interpret whether a transformation should be taken.

Order of differencing is determined based on stationarity testing of the presence of a unit root to identify the differencing order need for the Integrate part of the model. This can be done through Augmented Dickey Fuller test or the Kwiatkowski, Phillips, Schmidt and Shin test in order to decide if it is necessary for the data to be differenced which is further discussed in section *2.2.5*.

2. **The model parameters are estimated.**

Once a model has been identified, parameters can be estimated through Maximum Likelihood Estimation (MLE), which attempts to find the parameters which maximize the probability of obtaining the observed data.

3. **Candidate Model Comparison**

The candidate model selection of the candidate model based on the order of selected *AR* and *MA* terms can be accessed through Information Criteria to compare candidate models. The reiteration through this step allows for a comparison with formulated models. The two most popular in-sample measures used for model order selection through Akaike Information Criteria (AIC) and Bayesian Information Criteria (BIC).

**Akaike Information Criteria**

The Akaike Information Criteria is commonly used in the selection of the order of a model by rewarding higher goodness of fit and penalizing for complexity. For a single model it does not provide much information but in comparing models the lowest *AIC* model is the most suitable. This balance between the complexity and goodness of fit can be formulated as:

$$AIC = -2\log L(\theta) + 2k \tag{11}$$

The $L(\theta)$ is the likelihood of the candidate model, which is penalized by $k$ number of parameters of the candidate model.

**Bayesian Information Criteria**

The Bayesian Information Criteria also known as Schwarz Information Criteria (Schwarz, 1978) is another information criteria that is very closely related to the AIC set in the Bayesian context. The BIC penalizes the model complexity greater than the AIC which can be formulated as:

$$BIC = -2\log L(\theta) + k\log n \tag{12}$$

Where *the addition of n is for the*

4. **The model is ready for forecasting.**

The final step is conducting the prediction from the selected model to be analyzed across selected accuracy measures.

### 2.2.5 Stationarity

An important concept in time series analysis is the stationarity of time series, which says that its expectation and variance in the assumed data-generating process are constant across time $t$. We can separate this between the forms of strictly stationary or weak stationarity.

Strictly stationary series are those generated by a data-generating process that experiences no change in the time origin, also explained as the joint distribution of $n$ observations made at any set of time is the same for all sets of time.

Weakly stationary series are relaxed forms of strictly stationary series if it has mean and covariance stationarity. This means that the following assumptions must hold:

$$E[Y_t] = \mu_t = \mu \tag{13}$$

$$Var[Y_t] = E[Y_t - \mu_t] = \sigma_x^2 = \sigma \tag{14}$$

$$Cov[Y_t, Y_s] = E[(Y_t - \mu_t)(Y_s - \mu_s)] = \gamma(|s - t|) \tag{15}$$

For auto regressive moving average models, the assumptions are that the underlying data-generating process is weakly stationary.

**Stationarity Assessment**

Stationarity of the assumed data-generating process is an important aspect of time series analysis, which can help improve forecasting accuracy (van Greunen *et al.*, 2014). The first can be done through visual assessment of plotting the time series plot to assess if the variation and mean of the data is constant across time is constant. This is a very intuitive approach that allows for professional judgement of the time series to be made. Trends or seasonality can be further assessed from the patterns in the data.

Stationarity can be further assessed through statistical testing methods, of which the most common being the Augmented Dickey Fuller (ADF) test (Dickey and Fuller, 1979) and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test (Kwiatkowski *et al.*, 1992). These are the two most used stationarity testing statistical tests for time series modelling.

**Augmented Dickey-Fuller Test of the Unit Root**

Originally taken to test against the first order lags, the ADF test has been further augmented to capture higher order of lags across time series. The null hypothesis of this test is that there contains the presence of a unit root in the first order, implying the lack of stationarity in the data, but only for the test for difference stationarity without linear trend. Difference stationary processes containing a stochastic trend and are integrated of order one(Larsson, Clements and Hendry, 2001). The ADF test is based on the first order autoregressive process which can be explained as:

$$\Delta y_t = (\emptyset_1 - 1)Y_{t-1} + \sum_{i=1}^{p-1} Y_i \Delta Y_{t-i} + \varepsilon \tag{16}$$

We can further advance the model to incorporate drift with:

$$\Delta Y_t = \beta_0 + (\emptyset_1 - 1)y_{t-1} + \sum_{i=1}^{p-1} y_i \Delta y_{t-i} + \varepsilon \tag{17}$$

The model can then also be used to incorporate both the drift and the trend that tests stationarity around linear trend otherwise known as a trend stationary process. A trend stationary process being stationary around a deterministic function of time (Larsson, Clements and Hendry, 2001). The model for testing can be seen through:

$$\Delta Y_t = \beta_0 + \beta_1 t + (\emptyset_1 - 1)y_{t-1} + \sum_{i=1}^{p-1} y_i \Delta y_{t-i} + \varepsilon \tag{18}$$

**KPSS Test of Stationarity**

The Kwiatkowski, Phillips, Schmidt and Shin test (KPSS) test (Kwiatkowski *et al.*, 1992) for stationarity is another statistical test that test the stationarity of the time series. The difference being that unlike the ADF test which's' null hypothesis tests if $y_t$is integrated by order one *I(1),* the KPSS tests the null that the time series does not have a unit root and is non-stationary. This follows the tendency that most economic time series experience a unit root. The test assumes that the time series is stationary around a deterministic trend formulating the model:

$$y_t = d_t + r_t + \varepsilon_t \tag{18}$$

In which $d_t$ is the deterministic trend, $r_t$ is a random walk, and $\varepsilon_t$ is the stationary error term.

### 2.2.6 Autocorrelation

Autocorrelation is an important aspect of time series analysis, characterized as the dependence across time that makes time series data fascinating. It measures the linear relationship between different lagged values of a time series. We can calculate this for the $j^{th}$ lag and T is the length of a time series through the sample equation:

$$r_j = \frac{\sum_{t=j+1}^{T}(y_t - \bar{y})(y_{t-j} - \bar{y})}{\sum_{t=1}^{T}(y_t - \bar{y})^2} \tag{19}$$

We can assess the autocorrelation using the correlogram of the autocorrelation function known as the ACF, which plots the values of autocorrelation to be compared with the error bands. The Partial Autocorrelation Function (PACF) and its subsequent plot is another measure of autocorrelation of our data that considers the linear dependence with lag $k$, not accounting for the lags in between.

The Correlogram of the ACF and the PACF plots a bar with the value of the ACF and PACF between the values of 1 and -1 for each lag across the x-axis. The use of the ACF and PACF plots can help detect randomness in our data and further order of our Autoregressive process and moving average process within ARIMA models.

### 2.2.7 Automated Model Selection

Individual model selection can prove to be tedious and time consuming when many models need to be trained, which drives practitioners to use effective automatic approaches. One of the most popular implementation is the Hyndman-Khandakar algorithm (Hyndman and Khandakar, 2008) for automatic ARIMA model selection. This automatic approach follows much of a similar approach that manual model selection takes mentioned in the previous section *2.2.4*, but has intuitive software implementation for use in the R programming language in the forecast package (Hyndman *et al.*, 2024).

The algorithmic process begins by determining the order of differencing for the integrated component $d$ of the model. A repeated KPSS stationarity test is performed over the data, to determine a value $0 \leq d \leq 2$ that can be used within the model.

The next process in the algorithm is the determination of value of $p$, and $q$ by minimizing the in-sample information criteria using a stepwise search across a set of limited parameters avoiding an attempt to find a model across all the parameters. The first step in this process uses four initial models of *ARIMA(2,d,2)*, *ARIMA(0,d,0)*, *ARIMA(1,d,0)* and *ARIMA(0,d,1)*. The lowest information criteria model is taken, and its values are varied plus or minus one to find a best model to consider.

This algorithm provides a sensible process for estimating and training ARIMA models automatically but may not find the most optimal model. The application of such processes should be considered as to the needs of the modeler. If there is a requirement for a single or several models, an experienced practitioner may want to identify their own model for their own purposes, yet in the case where there are many models needed to be identified with time constraints of producing a forecast, automatic approaches may be more suitable.

## 2.3 Support Vector Regression

Support vector machines (SVM) were first introduced by (Cortes and Vapnik, 1995) which has its roots in statistical classification and pattern recognition problems. They have gone on to be extended to solve non-linear regression problems. The terminology for support vector machines, when used on classification problems, is most notably known as Support Vector Classification, whereas for regression problems, it is referred to as Support Vector Regression.

SVR models are grounded in its quadratic optimization technique, used to model a hyperplane within the original input space through non-linear mapping. The technique attempts (Ojemakinde, 2006) to leave as much distance from the closest observations while the observations in the maximum margin are referred to as support vectors. Using a symmetrical loss function to penalize values above or below a threshold, the objective is to find a function that has a maximum deviation ε from the actual target values where values beyond this boundary are not accepted.

Support vector regression models have been a powerful method in the world of machine learning. It has been used for many common regression-based problems but has also found grounding in time series forecasting due to its ability to recognize patterns within the data.

### 2.3.1 Background

Support vector machines can be formulated as quadratic optimization techniques in finding the optimal hyperplane(Awad and Khanna, 2015). SVR introduces an ε insensitive region along the function creating a tube the width of ε along the regression function given by:

$$f(x) = \langle w, x \rangle + b \tag{20}$$

Where $w$ refer to weights and $b$ is the bias which has $x$ mapped to a high dimension feature space through the dot product. We can write the objective function where $\|w\|$ is the magnitude of the normal vector as:

$$Min\ \frac{1}{2}\|w\|^2 \tag{21}$$

That is subject to the following constraints:

$$y_i - \langle w, x_i \rangle - b \leq \varepsilon \tag{22}$$

$$\langle w, x_i \rangle + b - y_i \leq \varepsilon \tag{23}$$

This quadratic optimization problem may be feasible for all ε but can have a soft margin (Smola and Schölkopf, 2004) which introduces non-negative slack variables $\zeta_i$ and $\zeta_i^*$ that can further formulate the problem as:

$$Min\ \frac{1}{2}\|w\|^2 + C\sum^{t}(\xi_i + \xi_i^*) \tag{24}$$

$$y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i \tag{25}$$

$$\langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i^* \tag{26}$$

Where C represents a cost function of determining the trade-off between allowance of deviations larger than the tolerated ε. The tolerated ε being the width of a "tube" in which allows for the penalization of values greater than its width. Only the values outside of the "tube" go towards the cost function We can see this visualized in the figure 4 where the regression function canters the "tube" which estimated width of ε is solved with the allowance of slack variables.

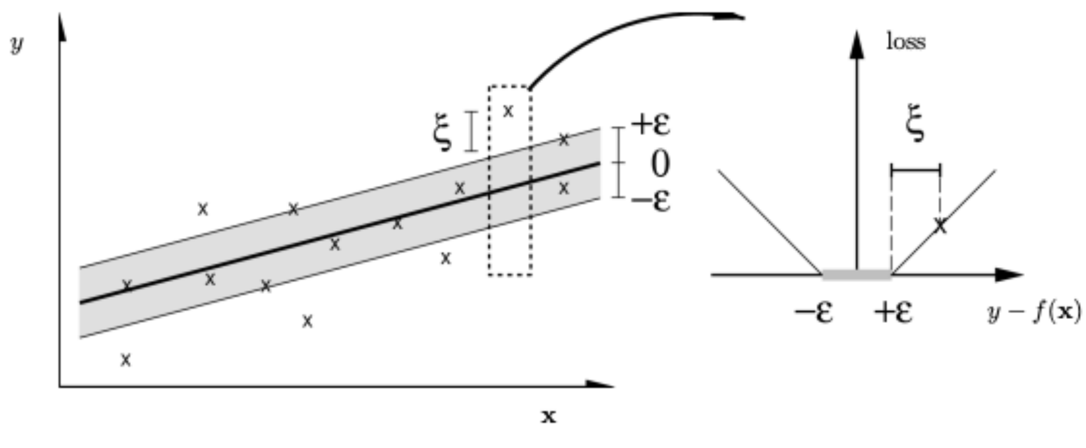


*Figure 4: Univariate Linear Support Vector Regression and Linear Loss Function(Scholkopf and Smola, 2002)*

The solution for this problem is then given in the following function where the kernel function is used to satisfy the Mercers Condition (Cherkassky and Ma, 2004) that the kernel function is both symmetrical and positive semi-definite. The sample points $\alpha_i$, $\alpha_i^*$ are the difference in positive for values within the tube.

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + b \tag{27}$$

### 2.3.2 Kernels

The use of kernel functions helps make the Support Vector algorithm map its hyperparameters in a higher dimensional space for non-linear modelling to achieve higher accuracy (Awad and Khanna, 2015). Kernels are heavily used across different machine learning techniques such as Neural Networks to support non-linearity. Some common kernel functions used for Support Vector Regression can be seen further.

**Linear**

The simplest form of kernel function can be seen as:

$$k(x_i, y_i) = x_i * y_i \tag{28}$$

**Polynomial**

$$k(x_i, y_i) = (1 + x_i * y_i) \tag{29}$$

**Radial Basis Function (RBF)**

The RBF being the most commonly used (Ojemakinde, 2006) is due to it its bounded value between 0 and 1 making it less computationally expensive than the polynomial function. This is also a commonly used kernel function used in neural network models(Cerqueira, Torgo and Mozetič, 2020) For the purpose of this analysis, the RBF kernel function will be used on the time series data within this study.

$$k(x_i, y_i) = e^{\left(-\frac{\|x_i - y_i\|^2}{2\sigma^2}\right)} \tag{30}$$

### 2.3.3 Hyperparameter Selection

While the kernel selection is an important input to modelling SVR, the parameter selection for the value of $C$ and ε are also vital in returning accurate predictions. The selection of the three parameters provides generalized performance due to the complication of the model complexity (Cherkassky and Ma, 2004). The user-defined parameters are typically either selected based on user expertise, which is subjective to the prior knowledge, or can be one through computationally expensive cross-validation.

The parameter C as an input to the objective function can be said to be the trade-off between the model complexity and a degree of tolerated deviation. Whereby the value of C has been noted in (Ahmed *et al.*, 2010) that it is not sensitive to predictive performance but manual determination of the parameter can be done based on the range of the output values, but due to the presence of outliers, it has been proposed by (Cherkassky and Ma, 2004) as the following equation.

$$C = \max(|\bar{y} + 3\sigma|, |\bar{y} - 3\sigma|) \tag{31}$$

The parameter ε is a control for the width of the ε-insensitive zone as seen in the figure 4. It is known to be proportional to the input noise level of the standard deviation but should consider the sample size of the training sample. The proposed method provides good performance that scales with large training sample sizes.

$$\varepsilon = 3\sigma\sqrt{\frac{\ln n}{n}} \tag{32}$$

While the use of cross-validation is computationally expensive, it can be performed if the modeler prefers to leave parameter selection out of subjectivity in which interpretation can be left to the in-sample loss function. Cross-validation can be performed to minimize the loss function using an accuracy measure such as RMSE. Loss functions quantify how well the model approximates the true value. Despite the setback in timeliness, the selection of the parameters can be achieved through many software packages, such as Caret in R (Kunh and Max, 2008).

The simplicity of the parameters makes the use of SVR model a very attractive tool for practitioners as top performers with ease of implementation.

## 2.4 Neural Network Approaches

One of the most popular methods for electricity pricing in recent years is the use of deep learning neural network-based approaches. Some of their major strengths are their flexibility and ability to model non-linear data. On the other hand this does not always result in more accurate point forecasts (Weron, 2014). Another benefit is the number of developing approaches which makes it difficult to find an approach that does not locate an optimal output. This also means thorough comparisons of models are difficult to be made. In this study, Long Short-Term Memory network models were performed due to their popularity in their success of electricity price forecasting.

Neural networks are built on the biological foundational architecture of how neurons in the brain function for learning algorithms. The Recurrent Neural Network (RNN) is a form of these modelling techniques which has a wide range of applications using sequential data such as language processing, image classification, and time series analysis. They have solved the issues of retaining information by allowing information to persist by passing information from one step to the next (Colah, 2015). One of the issues with this is that long term dependencies do not get picked up and a gradient vanishing or exploding problem arises as a down fall which lead to Long-Short Term Memory (LSTM) models proposed by (Hockreiter and Schmidhuber, 1997). This brings LSTM models as a preferable form of RNN.

### 2.4.1 Foundations of Neural Networks

Neural networks are simply defined (Gurney, 1997) as being an assembly of processing elements of units whose functionality is based on biological neurons, which has the process ability stored in the inter-unit connection weights obtained from learning from a set of training patterns. A neural network takes inputs and builds a non-linear function in order to predict a response variable. Built on layers, an example of a single layer feedforward network can be viewed in the Figure 5, where each layer of a model taking in four input features that feed into $K$ hidden units. The hidden layer of the example model in the Figure 5 computes an activation of $A_k = h_k(X)$ as a non-linear transformation of the inputs in the input layer. These activations are not observed and the function of $h_k$ are learned through the training of the model.

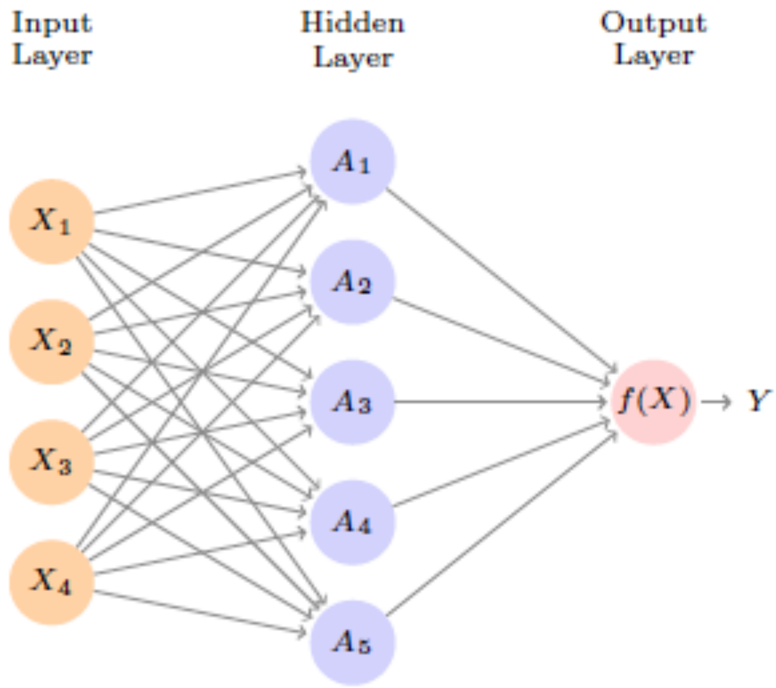


*Figure 5: Single Layer Feed Forward  Neural Network (James et al., 2023)*

One of the most common form of neural networks, shown in the Figure 5 are called feed forward network due to the fact that there are only forward connections between neurons, and can be modelled with the form:

$$f(x) = \beta_0 + \sum_{k=1}^{K} \beta_k g\left(w_{k0} + \sum\nolimits_{j=1}^{p} w_{kj} X_j\right) = \beta_0 + \sum_{k=1}^{K} \beta_k A_k \tag{33}$$

Where $K$ represents the number of hidden layers in the model, *g(.)* is the non-linear activation function. The variable $w_{kj}$ in this instance are the weights that determine the contribution of the input $X_j$ to the output. The similarities to a linear regression model if the activation function were reduced to a constant, are evident from such formulation and highlight the role of the activation function. This function is not only connected to the hidden layer (as represented in Figure 5) but also crucial for mapping the complex non-linearities in the data.

The fitting of a simple neural network takes the form of minimizing's the squared error as a loss function $f(x)$,which can be used in tuning the model through the form

$$\sum_{t=1}^{n} \left(y_t - f(x)\right)^2 \tag{34}$$

A key idea behind finding the weights of model is the concept of gradient descent, which i4s an algorithm used in the fitting of the model to search the hypothesis space for weights which that best fit the data. This provides the basis for backpropagation in which the errors are used to

The tuning a neural network can be done across a variety of hyperparameters depending on the size of the neural network's architecture. The hyperparameters are decided and set before the estimation of the model where the minimization of the lost functions can be compared to find the suitable model.

Some common hyperparameters (Bartz *et al.*, 2023) include the number of layers, the number of unit, and the number of epochs which refers to the number of iterations that the data will be fed through the model for training. As the values of these hyperparameters increase, so does the complexity of the model.

Another set of hyperparameters involve the non-linear activation function to be used within the layer. Some common non-linear activation functions used in machine learning (Dubey, Singh and Chaudhuri, 2022) are:

**Sigmoid Activation function**

$$g(z) = \frac{e^z}{1 - e^z} = \frac{1}{1 + e^{-z}} \tag{35}$$

**Tanh**

$$g(z) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{36}$$

**Rectified Linear Unites (ReLU)**

$$g(z) = max(0, z) = \begin{cases} 0 \; if \; z < 0 \\ z \; otherwise \end{cases} \tag{37}$$

### 2.4.2 Recurrent Neural networks

Recurrent Neural Networks (RNN) have slightly different architecture than the single layer mode presented previously as they allow loops for information to be persisted and passed on between time steps. A RNN, in its simplest form with a single hidden layer as Figure 5, consists of the input layer, recurrent hidden layer and the output layer. As seen in the figure

6, the looping can be unrolled to visualize the process across 3 time steps, where $A$ is the activation function in the hidden layer at time $t$ seen here with a Tanh function, but also commonly set with a Sigmoid function. The $X_t$ is the input in time $t$, while $h_t$ is the output in time $t$. While the un-rolled diagram for the looping occurring in the network appears sequentially through time the hidden layer is a hyperparameter selected, in which this case this is a simple network with a single layer.

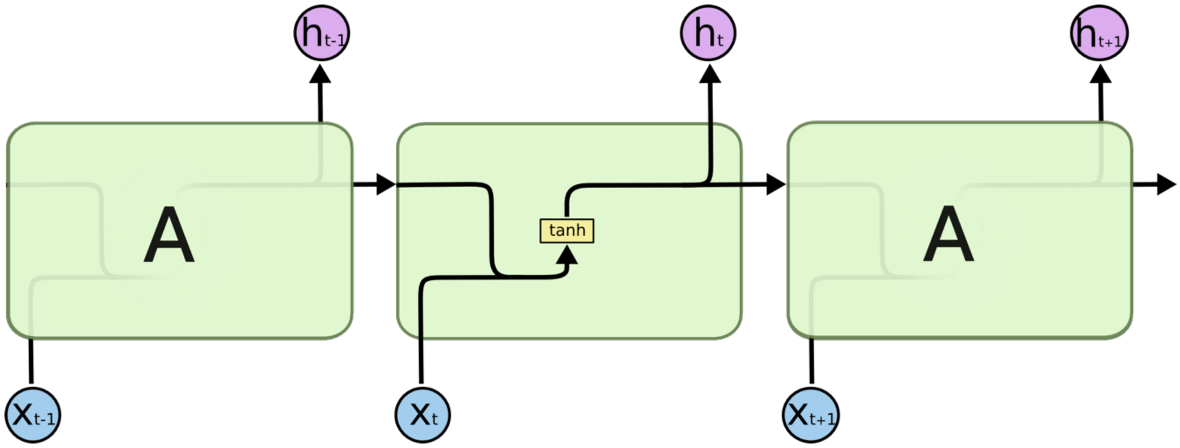


*Figure 6: Single Layer Recurrent Neural Network (Colah, 2015)*

The process by which information (Schmidt, 2019) is passed across between layers uses a weight matrix $W_{xh}$ for information passing into and out of the hidden layer. Also used is a hidden-to-hidden weight matrix of $W_{hh}$ used to pass information between time steps in the hidden state, and the bias parameter $b_h$ which allow us to formulate our hidden variable $A_t$ and output variable $h_t$ as:

$$A_t = g(X_t W_{xh} + A_{t-1} W_{hh} + b_h) \tag{38}$$

$$h_t = g(A_t W_{ho} + b_h) \tag{39}$$

A drawback of Recurrent Neural Networks is their difficulty of handling long term dependencies through the vanishing gradient problem. Whereby through backpropagation through time, the matrix multiplication performed in backpropagation through time over very long sequences causing their gradient to decrease in each layer until it vanishes (Hyndman *et al.*, 2024).

### 2.4.3 Long-Short Term Memory Model

Long-Short Term Memory models were introduced as alternative form of RNN that are able to deal with the vanishing gradient problem (Hockreiter and Schmidhuber, 1997). The architecture gated cells to store more information outside of the traditional architecture previously explored (Schmidt, 2019). The gates provide a way for information to be optionally let into the model(Colah, 2015)

The architecture introduces three gates using sigmoid functions in the hidden layer consisting of the Input gate $I_t$, Output gate $O_t$, and the Forget gate $F_t$. We can see the graphical description of the gates in the Figure 7 as an extension of the architecture of a simple RNN in the Figure 6.

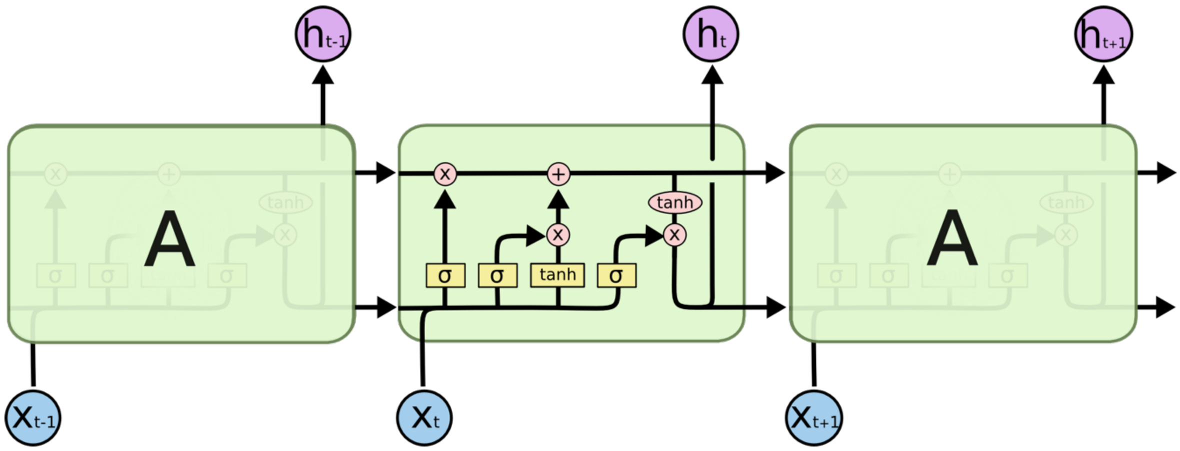


*Figure 7: LSTM Layers Architecture (Colah, 2015)*

The first step in the flow of information is the selection of data to keep in the Forget gate looking for values with a value of 1 to keep and 0 to remove. This gate takes the form of:

$$F_t = \sigma\left(X_t W_{xf} + H_{t-1} W_{hf} + b_f\right) \tag{40}$$

The second step in the flow of information is deciding what information will be stored from the Input gate that combines with the Candidate memory cell $\tilde{C}_t$ using a tanh activation function to create an update to the state, which takes the form as:

$$I_t = \sigma(X_t W_{xi} + H_{t-1} W_{hi} + b_i) \tag{41}$$

$$\tilde{C}_t = \tanh(X_t W_{xc} + H_{t-1} W_{hc} + b_c) \tag{42}$$

The third step in the process is to update the old memory content $C_{t-1}$ into the new memory state $C_t$ using the gates in the previous steps within the flow of information. This operation is done using the element wise operation in the form

$$C_t = F_t * C_{t-1} + I_t * \tilde{C}_t \tag{43}$$

The final step is the computation of the output layer and the hidden layer made up of the gates introduced to the LSTM architecture

$$h_t = \sigma(X_t W_{xo} + H_{t-1} W_{ho} + b_o) \tag{44}$$

$$H_t = h_t * \tanh(C_t) \tag{45}$$

## 2.5 Alternative Methods to Electricity Price Forecasting

While the practical aspect of this paper employs modelling techniques from the families of Statistical and Computational learning models to forecast electricity prices using univariate series, there are alternative approaches that should be mentioned for further analysis. These approaches employ alternative methods, which mostly utilize vast amounts of data to forecast or construct a price based on different sets of needs. Their comparative results may be different than those gained through methods focused on univariate time series forecasts from learning from trends in the data.

**Multi-Agent models**

Multi-agent models are based on the methods used to forecast electricity prices before market liberalization, where demand and supply estimates are matched based on planned generation to derive a price based on known generation cost. These are known as production cost models, yet due to the inclusion of other actors and the rise of bidding practices in liberal markets, game theoretic approaches were developed to address market behaviour and competition in Electricity bidding structure. Utilizing multi-agent modelling approaches helps provide qualitative understandings of the direction of price but comes under the stress of requiring many assumptions to be embedded in the models surrounding market players, their strategy and how they interact.

**Fundamental Models**

Fundamental models try to capture the economic relationships underpinning the production and trading of electricity. They are commonly used as hybrid approaches with Statistical or Computational Intelligence models. They use fundamental drivers with inputs ranging from load, fuel prices, and weather data. They perform well for risk management and derivative management, yet some challenges in applying these methods include the access to data, which is dependent on markets, along with the difficulty of incorporating fluctuations in the fundamental drivers.

**Reduced Form Models**

Another forecasting approach to electricity prices is the Reduced form approach, which is used to replicate the characteristics of daily prices for derivative pricing and risk management. The two most common types of models under this grouping are Jump Diffusion models and Markov regime-switching models. The approaches separate between deriving the properties of the spot market, which this paper covers, and the forward market, which reduced form models excel at.

# 3 Results

## 3.1 Data Preparation and Visualization

As an essential step in time series analysis, the plotting of the total time series can be seen in Figure 1. The time series plot provides us with a visual guide by which we can make initial decisions about patterns and behaviour in the data. There are five years of daily prices resulting in a time series consisting of 1826 observations. Clear developments in the cost of electricity in the Czech Republic show a gradual upward trend beginning in the middle of 20221 and the growing volatility of the prices following the beginning of the war in Ukraine, which started on February 24$^{th}$, 2022.

A general interpretation can be made about the data from the price plot, such as that the data-generating process of electricity prices previously experienced a constant non-zero mean and homoscedastic tendencies during the calmer periods in comparison to the growing ever-volatile energy market before the second half of 2021. The volatility of the prices as a commodity carefully managed in supply and demand through the balancing act that systems operators perform, which means that the possibility of negative prices at specific time can occur. This may underpin the type of transformations that are commonly taken in financial markets such as the common use of returns.

It should be noted by the outstanding effect that the beginning of the Russian war, which began February 21, 2021, caused a very evident increase in the volatility and the magnitude of prices. Yet prices had begun to increase well before roughly increasing in the middle of the year. In Figure 8 it is evident that the sharp increase after the second half of 2021 before the onset of the war marked by the vertical plotted line. This is described by rising natural gas prices due to shortages in Asian markets affecting European spot prices for natural gas (Benecká and Hošek, 2022), which has a strong link to electricity prices. This volatility provides interesting challenges to forecasting and the need for certainty through the instability of the international situation.

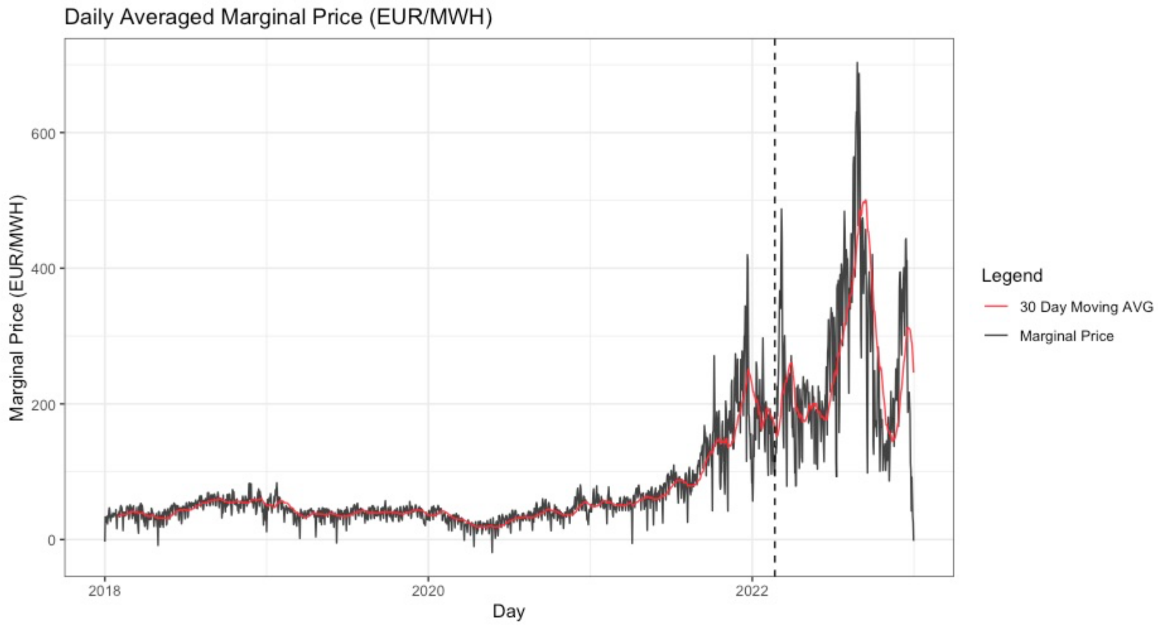


*Figure 8: Plot of the Daily Marginal Price and 30-day Moving Average (2018-2022)*

We can also interpret possible seasonality from the simple plotting of our time series data. There appears to be annualized seasonality, which can be described as a decline through the spring, leading up through the summer, and into the fall. Prices then peak going into the wintertime. We can see a resemblance of this seasonal development in Figure 9, where prices are plotted before the increase in prices and higher volatility. These developments follow some basic understanding about how electricity is demanded throughout the year. The standard inputs that influence electricity prices as described by the Energy Information Agency (EIA) are clear in that temperature changes throughout the year along with energy resources inputs to generation play a key role in influencing the price of electricity.

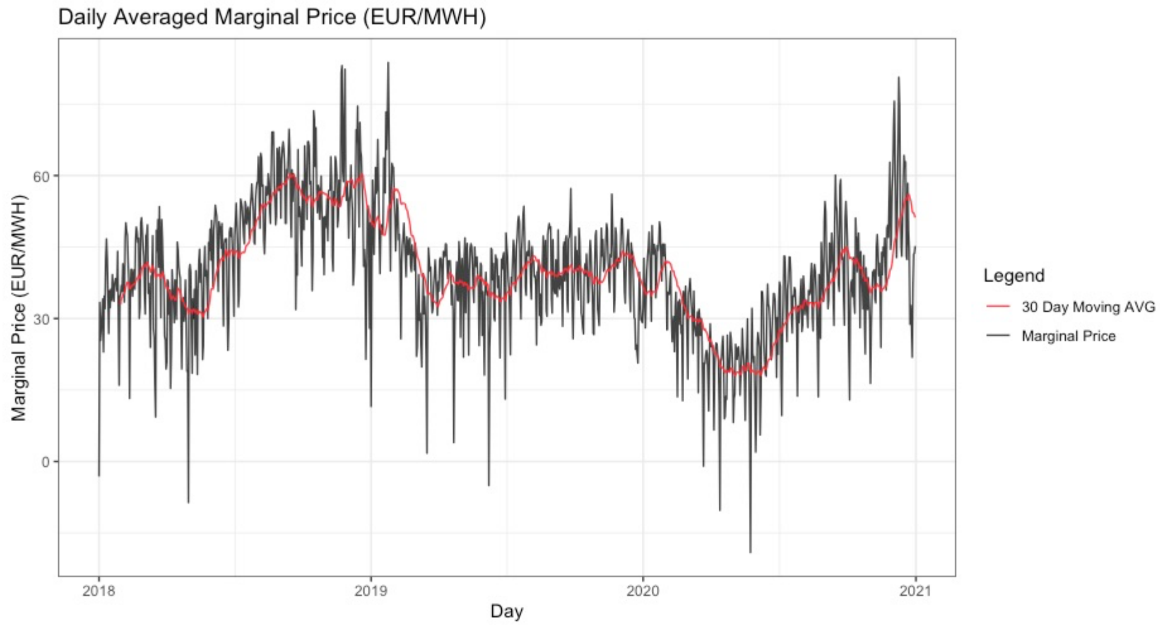


*Figure 9: Plot of Daily Marginal Price and 30-day Moving Average (2018-2020)*

### 3.1.1 Transformation

The data in question has been transformed using the log method. While there are instances of negative values within the data, this requires the use of a constant set to the minimum value of the data multiplied by two to bring the values of the time series away from zero. Forecasting is done using both un-transformed data and transformed for reference to the effect of such variance reduction techniques.

The transformed data can be seen in Figure 10, while using a log transformation is a approach used in time series analysis (Hyndman and Athanasopoulos, 2018), which has a benefit that the values are relative to the actual values, but the presence of negative values poses a challenge when transforming electricity price data.

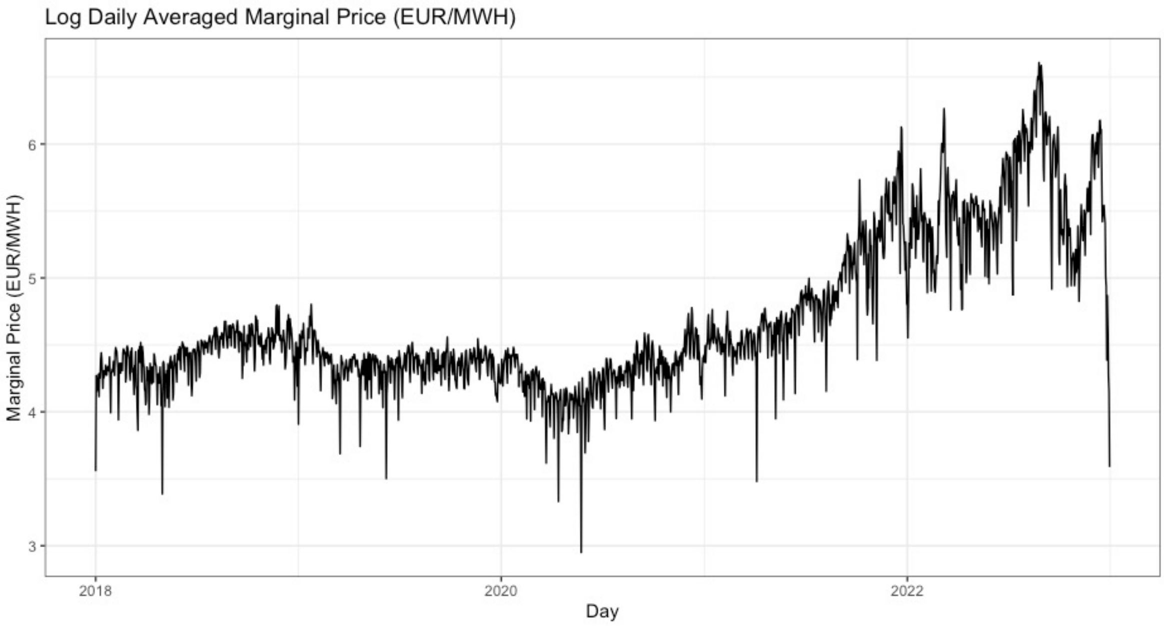


*Figure 10: Log Transformed Marginal Price per MWH*

Taking the log transformation is a transformation that is usually used for rescaling sample observations to fulfill assumptions (Curran-Everett, 2018) about the variability and to help distribute observations to achieve normality. In our case the taken log transformation is used to reduce variability in the data in which electricity prices are known to experience frequent shocks as the magnitude of these change over time. In order to take the natural log with negative values the data is shifted through the use of a constant calculated as two times the minimum value, which the operation can see in *(46)*.

$$\log(y_t - \min(y_t) * 2) \tag{46}$$

The transformation back to the electricity prices can then be done for comparable results produced without the transformation through,

$$e^{y_t} + \min(y_t) * 2 \tag{47}$$

### 3.1.2 Forecasting Horizons

The data that will be used follows two different time horizons, using a fixed-width rolling interval method to predict the next-day price of electricity and the next months. For the short horizon, 365 intervals will be used to forecast each day of the year, while the long-term horizon shall perform 12 forecasts across the year. The data is separated with a training window of four years that will roll over a single year (2022), ensuring that out-sample values of the previous window are included in the following windows model. The accuracy measures are then averaged over the length of all the windows. The training set and the values to be tested for the single-day prediction to be rolled over can be seen in the log-transformed, Figure 11 with the black plotted line representing the initial training interval. This fixed interval will predict the next day, shifting each attempt one day to the right until an entire year has been forecasted.

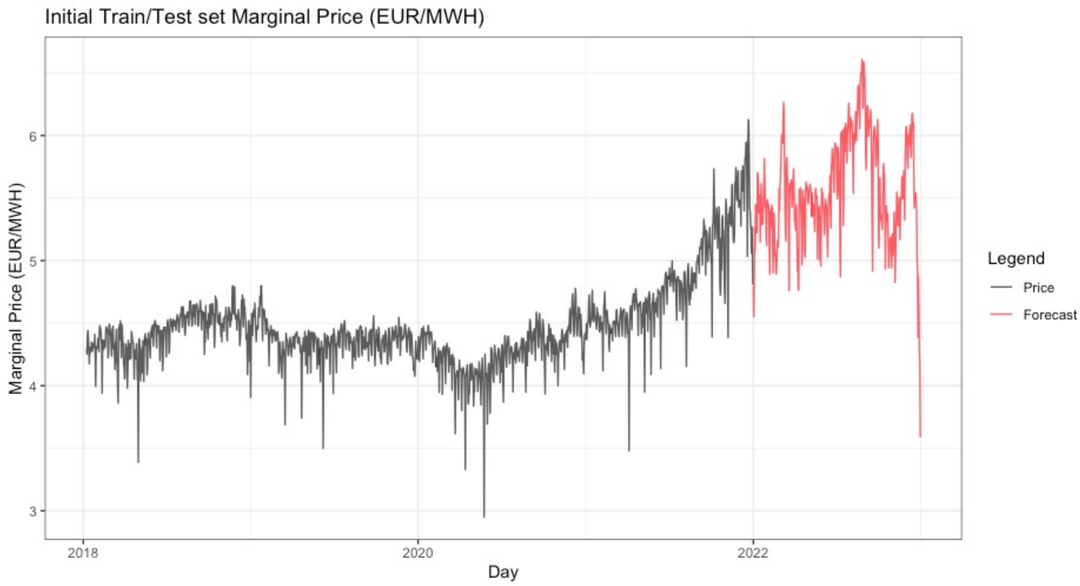


*Figure 11: Initial Training set with the test values to be rolled over in each interval.*

The long-term multistep forecasting horizons generally hold worse results than single-step forecasts. However, it is still an approach taken to find some clarity to the future as practical. The rolling interval method is used again for the long-term forecast for an accurate comparison across different time windows. The windows used are a four-year training window that rolls over 12 times for each month, allowing for train and test segments to evaluate the tendencies of the data is displayed in the segmented months in Figure 12. For the long-term forecast, a multi-input, multi-output technique will be employed to train a single model in order to forecast against the whole month of our lagged input data.

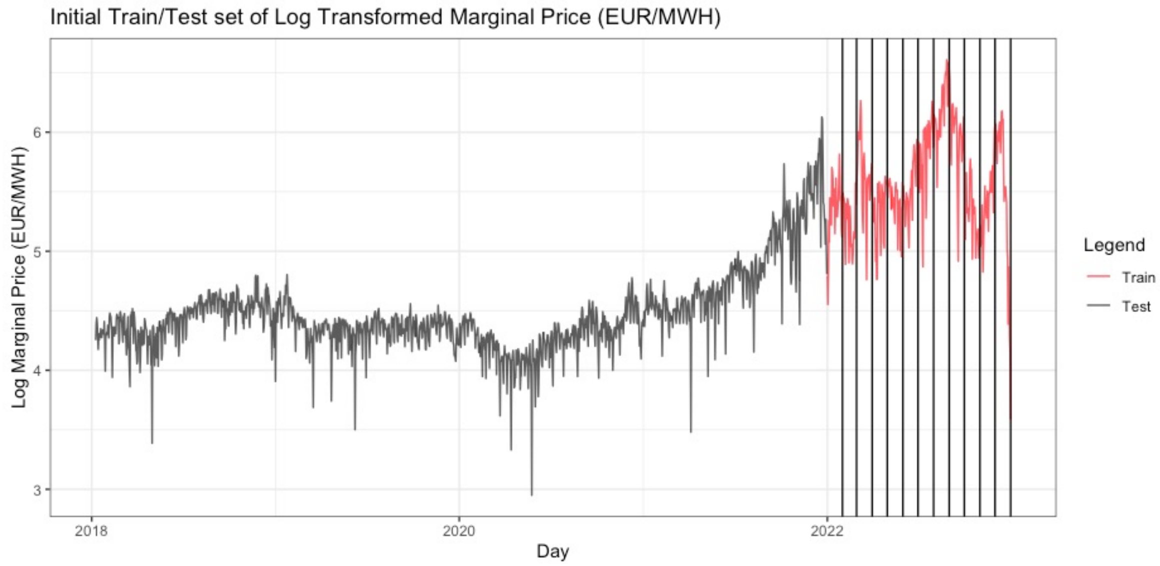


*Figure 12: Train/Test Rolling window for the Long-Term Multi Step Forecast*

Forecasting horizons and predictions tend to bear subjectivity. Still, the choice to cover a rolling interval over a year in both the short-term and the long-term rolling interval is to provides results across all-time series calendar information and its performance through varying periods of volatility is recommended as good practice for forecasting electricity prices by (Jędrzejewski *et al.*, 2022). The two different forecasting lengths between the single step and the month multistep, utilize the same length of their train set to maintain consistency. It is also good practice re-estimate on each time interval to arrive at accurate predictions that would follow the realistic use of the forecast.

## 3.2 Forecasting Process

### 3.2.1 ARIMA Forecast

In this section, the steps to formulating an ARIMA prediction will be taken for interpretation for the first interval. The manual process of identifying a model does not suit this situation of the rolling interval across an entire year, as it would require the identification of 365 different models, that would prove painstakingly time-consuming. For this reason, the Hyndman-Khandakar algorithm is used with implementation coming from the R forecast package (Hyndman *et al.*, 2024).

In identifying the correct order of differencing from the plotted log-transformed data, we can see a slight tendency for the series to rise in 2021 as the magnitude of change increases. We can use the KPSS test for stationarity, as seen in the table of its critical values against the test statistics in Table 1. Based on the results of the KPSS test, we would not reject the null and assume that the data is not stationary around linear trend. We would then consider differencing our log price.

| | Critical Value 5% | Test Statistic |
|---|---|---|
| **KPSS Test** | **0.463** | **5.937** |

*Table 1: KPSS Test of the First Training Interval (2018-01-01 to 2021-12-31)*

From the ACF and PACF in Figure 13, it is evident that serial autocorrelation that further supports the need for the data to be differenced for our ARIMA model. It is evident from the differenced ACF and PACF that there appears to be exponentially decreasing, representing the presence of an MA and AR process in the data.

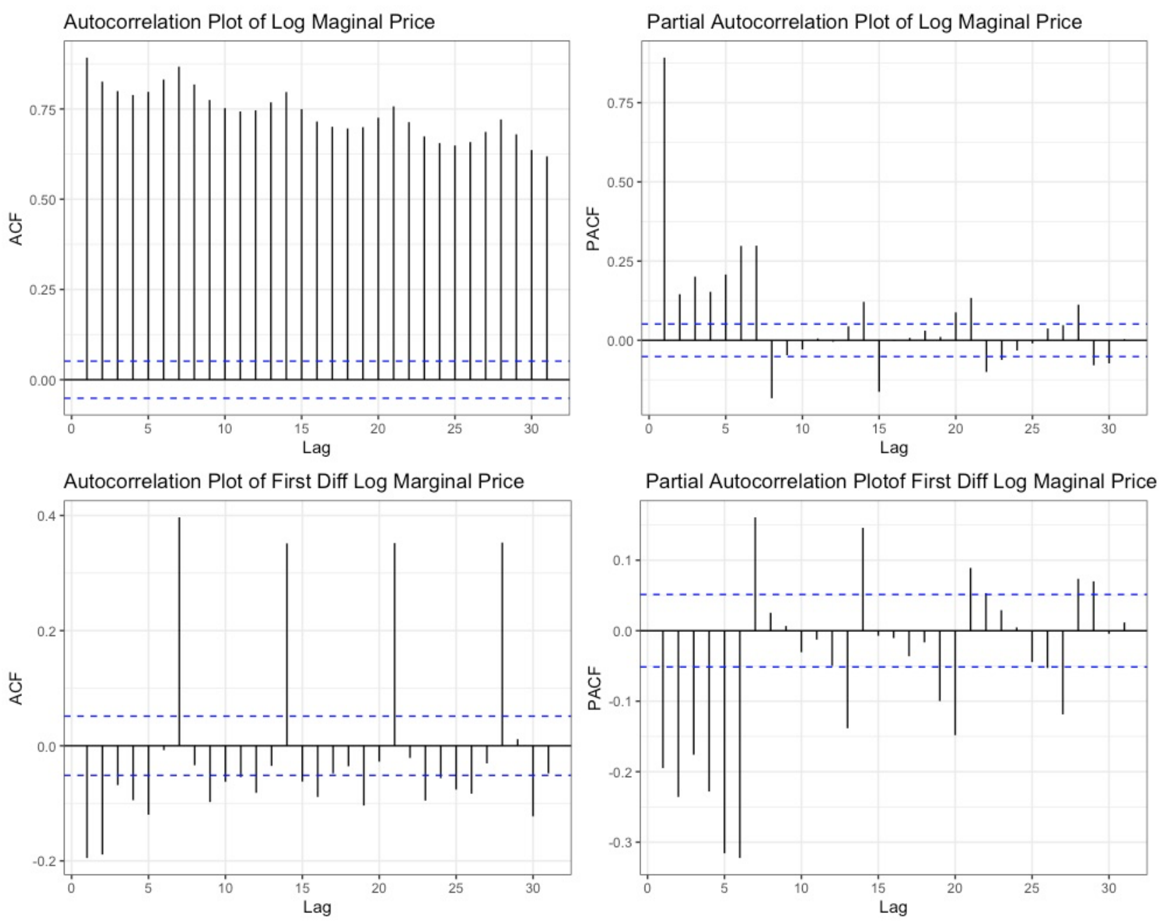


*Figure 13: ACF and PACF of the Log Price and its First Difference of the first Training Interval (2018-01-01 to 2021-12-31)*

We will not identify each model manually for each individual interval. Still, we will use a more automated approach to model selection with the development of the information criteria across all the time intervals for both the short-term forecast and the long-term forecast plotted in Figure 14 and 15. There is a clear difference in the use of the transformation to minimize the information criteria across the periods. However it is interesting to note the jumps in the information criteria for the short-term forecast with the log-transformation, as well as the increasing value as the year progresses. These jumps are not apparent between in the Figure 15 where the log transformation was not used, but have a more smoothed tendency.

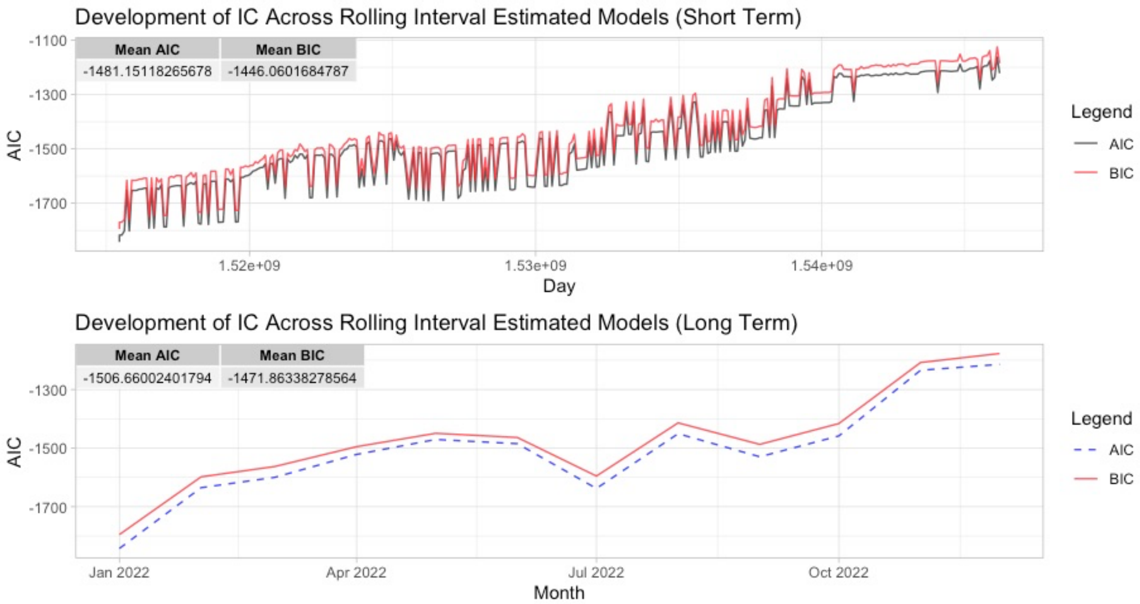


*Figure 14: Development of the Information Criteria Across Short-Term and Long-Term Intervals (log transformed)*

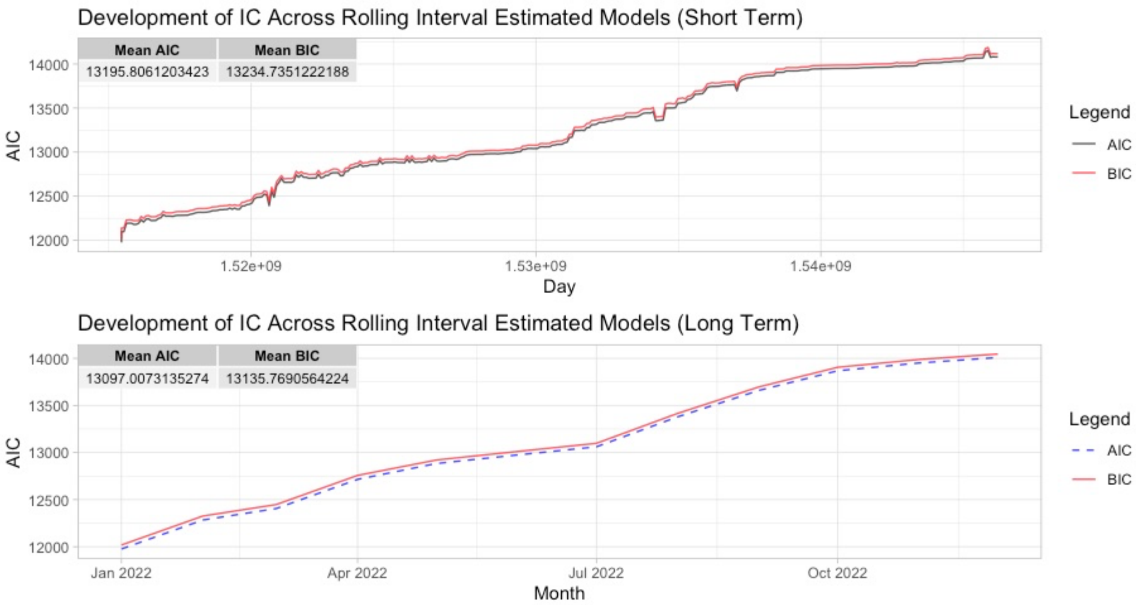


*Figure 15: Development of the Information Criteria Across Short-Term and Long-Term Intervals (without log transformation)*

### 3.2.2 Support Vector Regression Forecast

Model selection of the support vector machines can be tuned via cross-validation of the RMSE given a selection of hyperparameters to tune over. Support Vector Machine models require embedding dimensions that can be used to map our data to higher dimensional space, which, for the case of time series, is done by using the input of a lagged series of the day ahead price.

To capture linear and non-linear mappings of the data, we will use the radial basis function or, otherwise known as the Gaussian kernel for our model. This is a more general approach used on time series data, such as in (Ojemakinde, 2006) and in other industry-related forecasting problems (Ighravwe and Mashao, 2020).

The cross-validation performance can be expressed through the plot of the in-sample RMSE across the difference hyperparameters. As the rolling interval contains many models to be run. Figure 9 represents the performance of the first training interval lost function. The provided range of hyperparameters across sigma are given values of 0.001, 0.01, and 0.1 for sigma, and for the cost value tested, values were 1 to 10, 50, and 100.

A minimal value for the first interval is found for the log-transformed data at the point where ε it is equal to 0.1, and C is equal to 10, while for the un-transformed prices the values are 0.01 and 50. We can see the plotted values and their in-sample RMS across the different parameters in Figures 16 and 17.

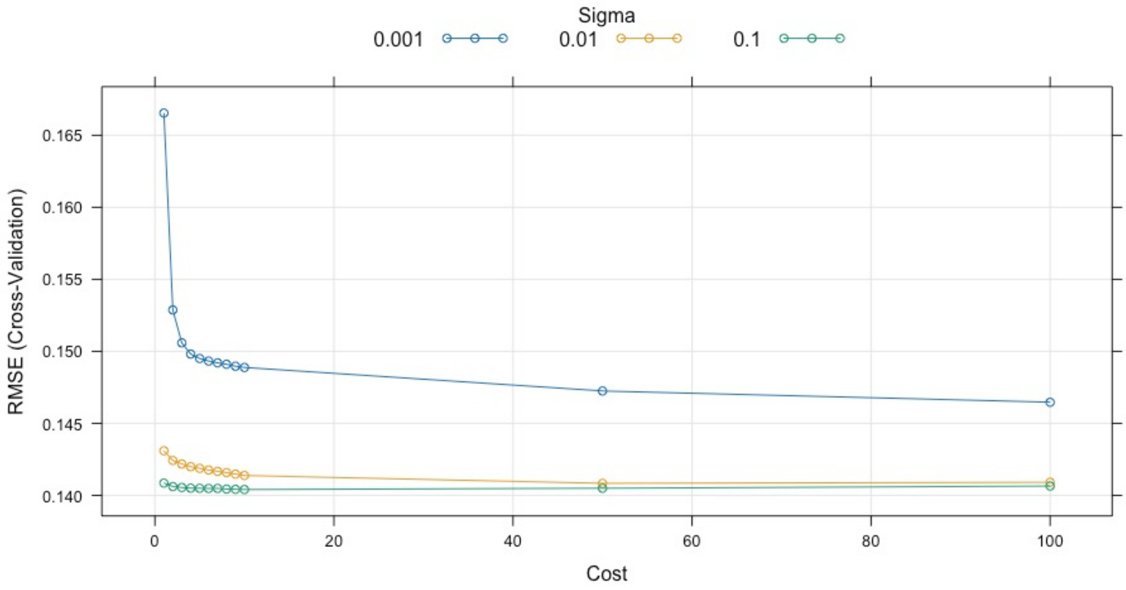


*Figure 16: Cross-Validation Performance on First Interval of Log Transformed SVR Hyperparameters*

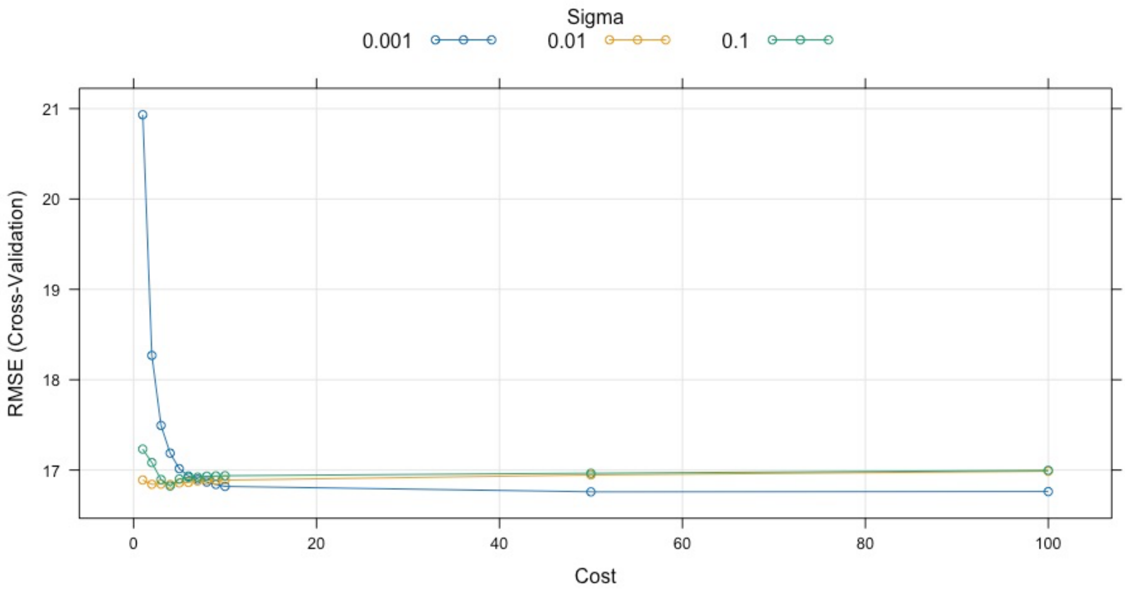


*Figure 17: Cross-Validation Performance on First Interval SVR Hyperparameters*

### 3.2.3 LSTM Forecast

An essential component of modelling with LSTM neural networks is the need for data preparation to normalize the data. The normalization ensures that the range of the inputs is set within a range of zero and one. This transformation is a recommended approach in using neural networks for time series data (Petneházi, 2018). Furthermore, the standardizing of the data supports the speed of finding an optimal solution to improve the efficiency of estimation (Peiqi, 2022). Standardizing the data in this way was done using minimum and maximum values returning values with a range of zero and one. This transformation can be understood through the following

$$y = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{47}$$

For this analysis, the modelling approach has been kept simple, containing a single LSTM layer. The hyperparameter tuning of the model is an extremely important step in building a quality LSTM model across different hyperparameters. The method used to tune the model hyperparameters was the grid search method across a variety of given hyperparameters. The given range of values for the number of units was 16, 36, 64, and 100, and the number of epochs used was 1,5 and 10. The number of possible combinations of models to be tested amounts to 12.

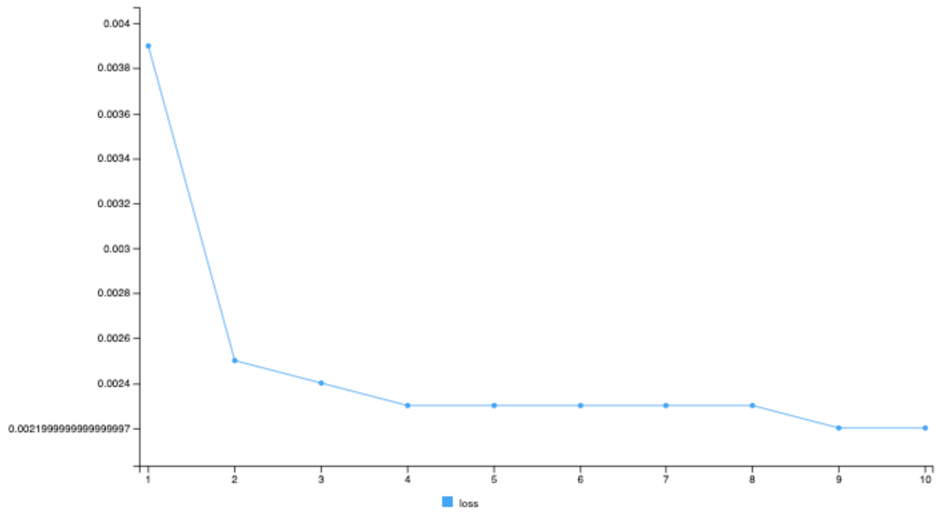


*Figure 18: Loss Function of Best Tune Hyperparameters (units = 16, epochs = 10) (log transformed)*

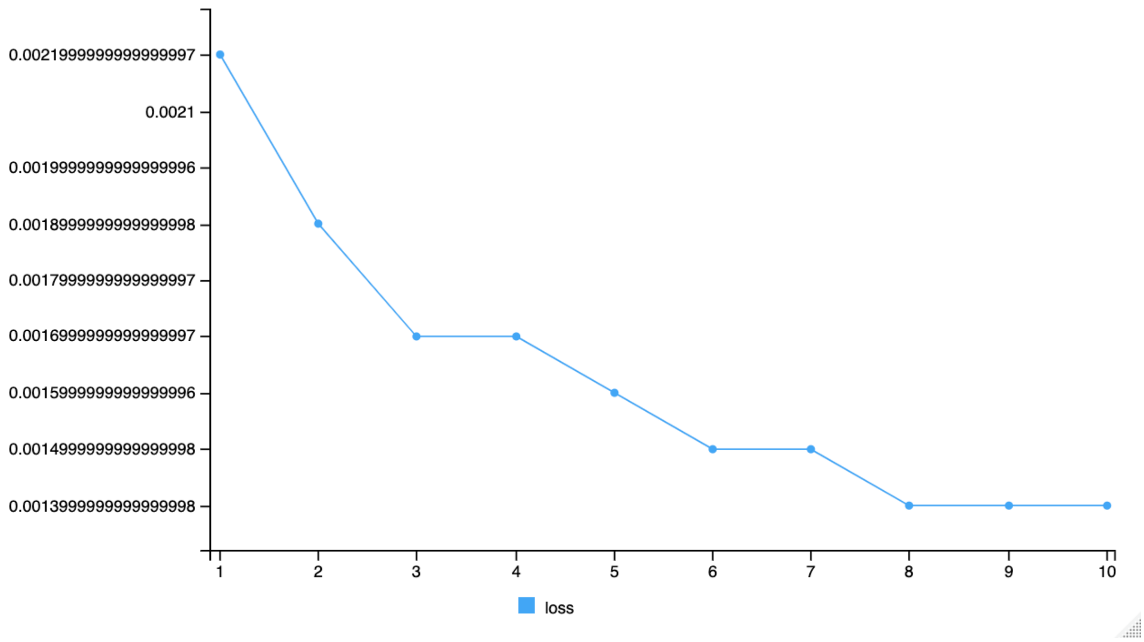


*Figure 19: Loss Function of Best Tune Hyperparameters (units = 36, epochs = 10) (Without transformation)*

Forecasts were performed using the Keras (Allaire and Chollet, 2023) and Tensorflow (Allaire and Tang, 2024) modelling packages in R. A pivotal attribute to performing predictions in this approach using Keras is that the package applies a 'modify-in-place' approach to the objects in in the global memory of the R or Python environment. This means that modelling with these packages applies a connection to any objects in memory by pointing to specific locations in which the model has connected with. Looping through intervals requires a careful approach to separating objects and generating individual objects in which prediction results are stored to ensure that each model is not retaining information from the first pass-through and forecast. The method used for getting around this issue is included in the Appendices Code 5.

## 3.3 Short Term Results

Comparing the three modelling approaches in terms of their ability to forecast, we can interpret the performance of the models based on the average out-sample accuracy measures and the visual performance of the forecasts across all time steps. The rolling interval method allows us to see the performance across large, unexpected spikes, which are very prevalent in the data. This can be seen in the significant jump in the spring and in the prediction for the decline at the end of the year, that were evident across the initially plotted values.

We can see from the linked forecasted values against their test values across all the intervals for each individual model across 2023 in Figures 19 through to 24 below. An interesting note is that from a visual assessment of the performance, the SVR modelling technique achieved a very close accuracy across large spikes proving visually to be the best performer. The ARIMA and LSTM approaches struggled with these spikes and proved to be more comparable among themselves. Comparably ARIMA and LSTM models appear to perform better having not underwent a log transformation.

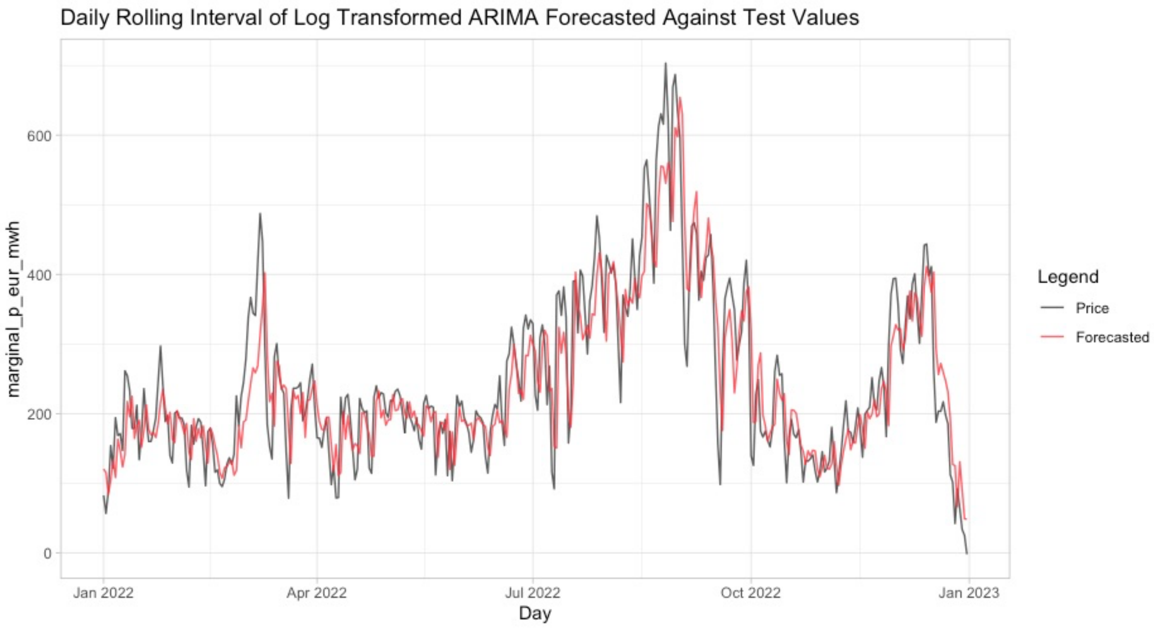


*Figure 20: ARIMA Next Day Forecasted Price Against Test Values (Log transformation)*

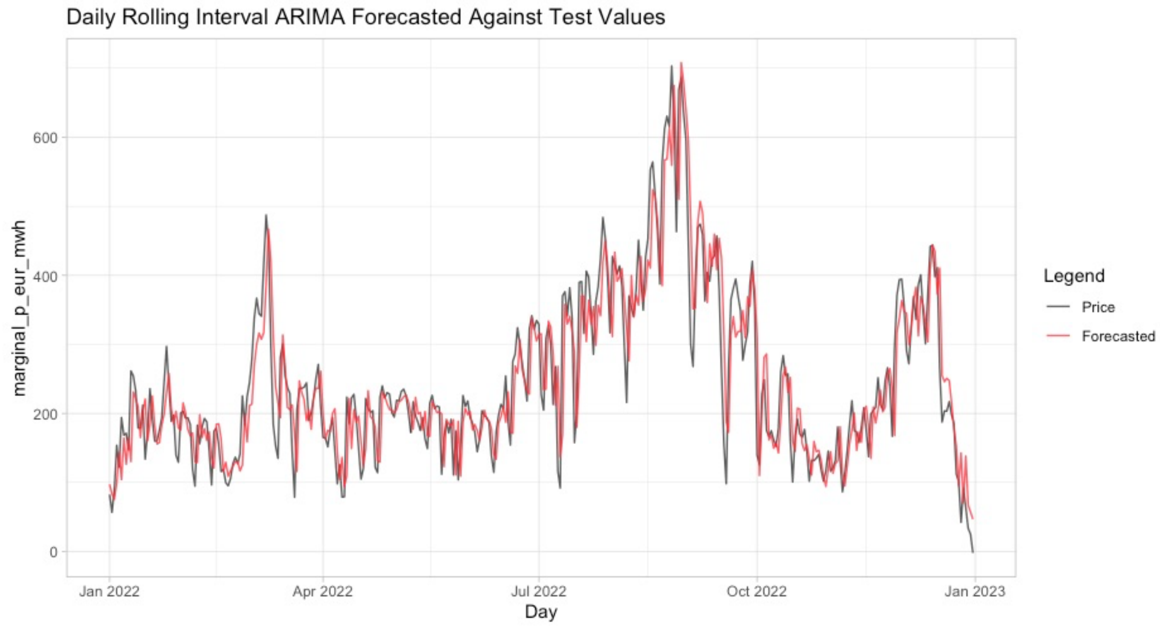


*Figure 21: ARIMA Next Day Forecasted Price Against Test Values (without transformation)*

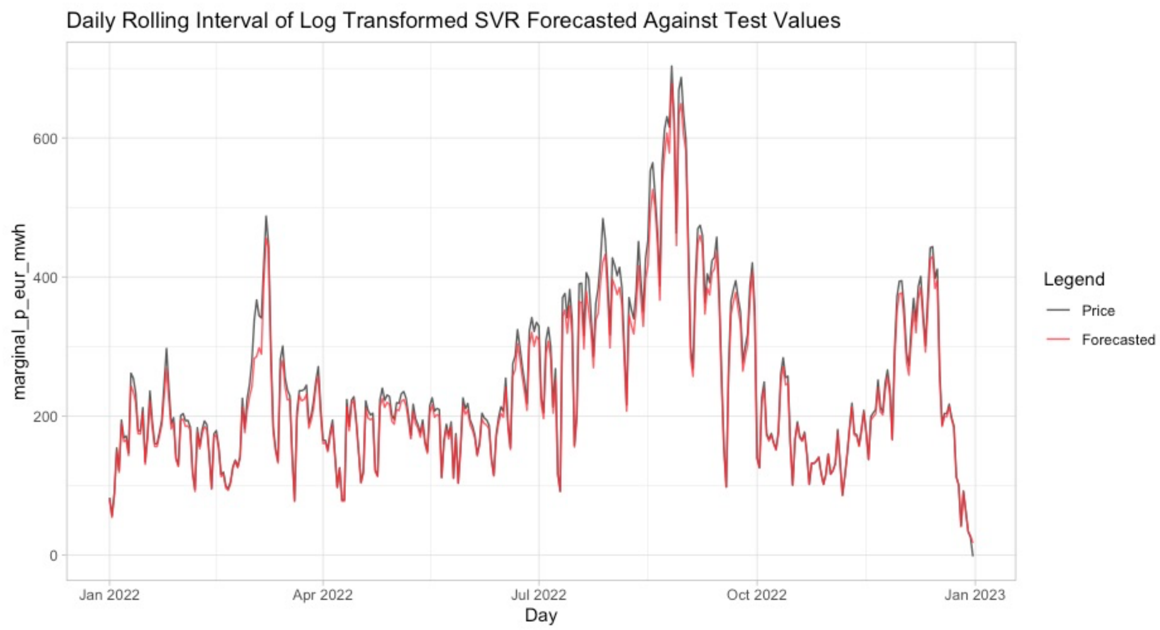


*Figure 22: SVR Next Day Forecasted Price Against Test Values (log transformation)*

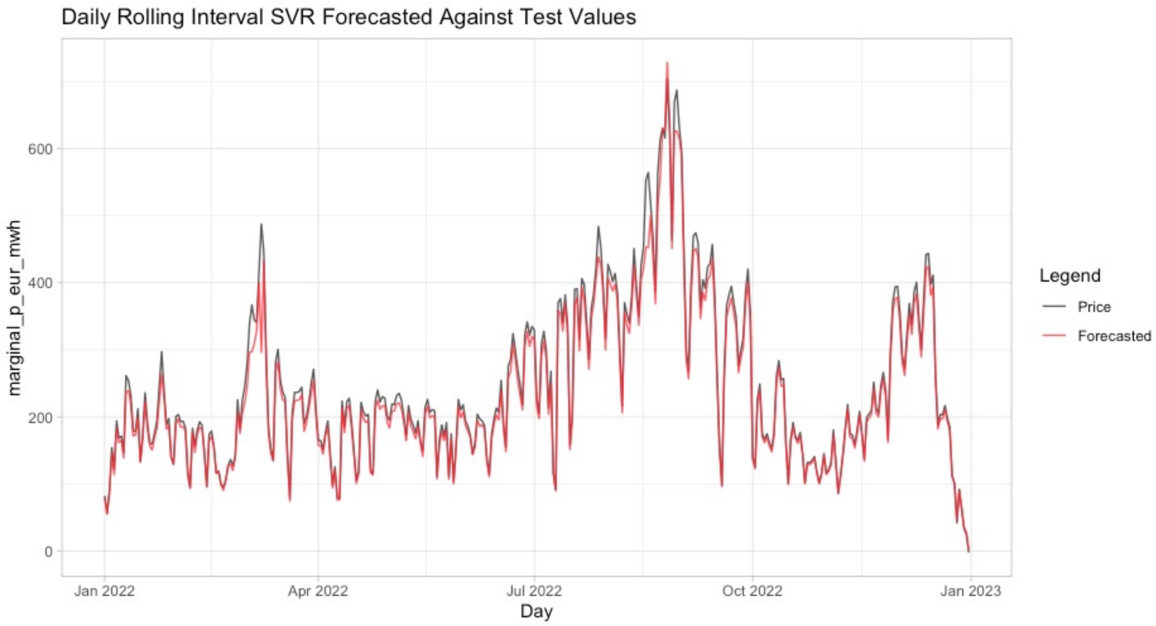


*Figure 23: SVR Next Day Forecasted Price Against Test Values (Without transformation)*

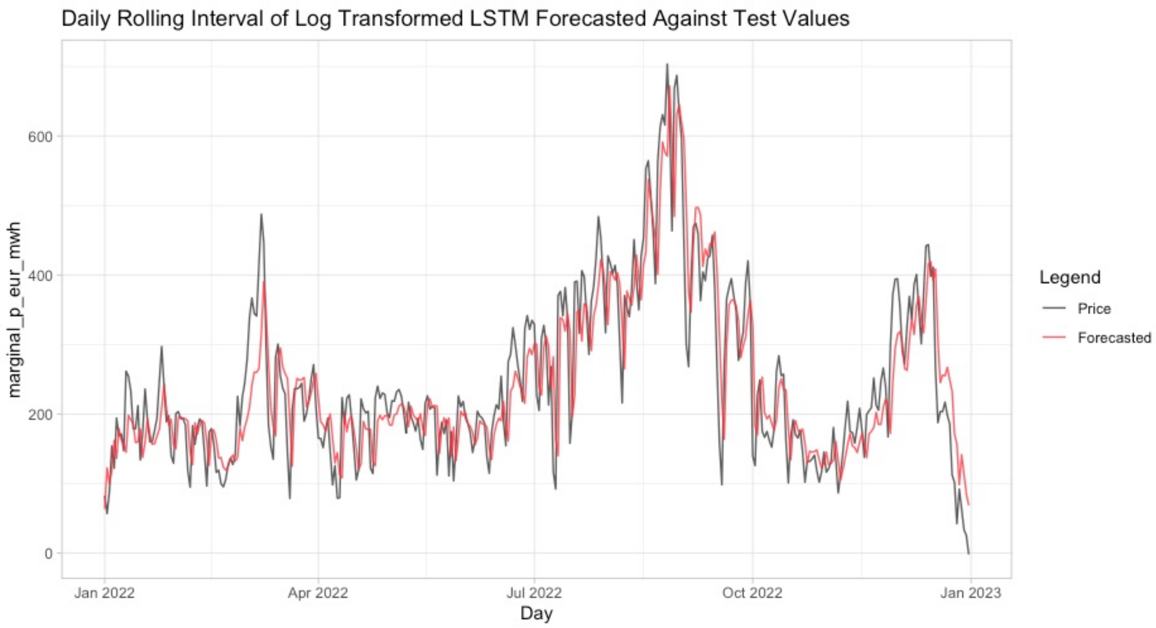


*Figure 24: LSTM Next Day Forecasted Price Against Test Values (Log transformation)*

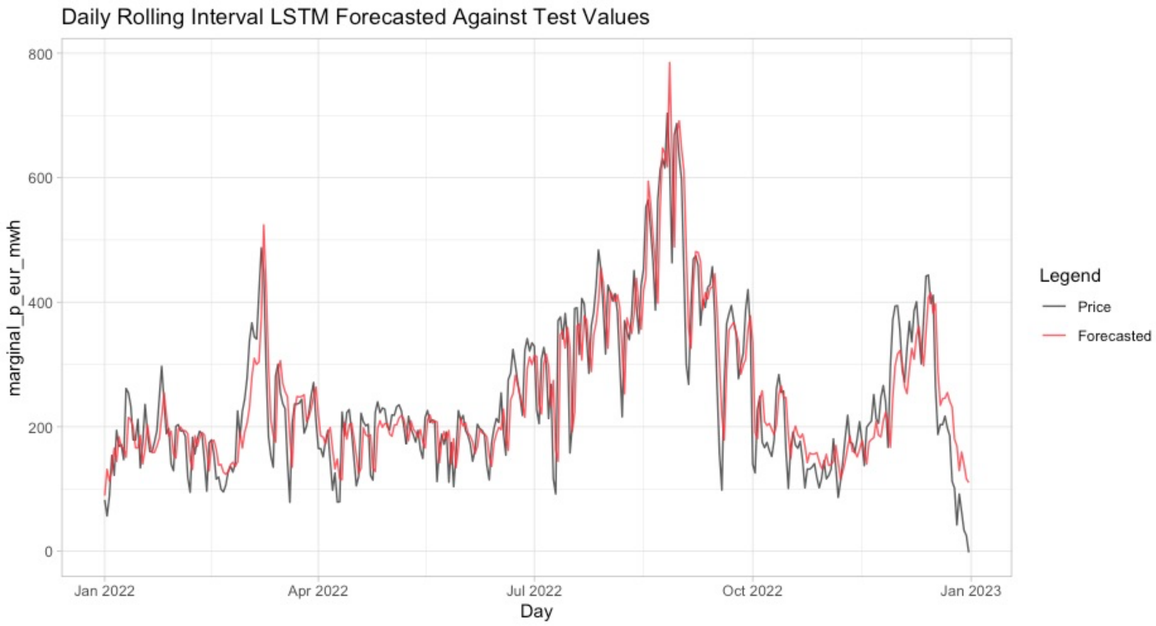


*Figure 25: LSTM Next Day Forecasted Price Against Test Values (Without transformation)*

The SVR model performs the best in predicting the daily price accurately through the greater swings in price, where its mean RMSE and sMAPE across the year, which are shown through annual average accuracy measures in Table 2. On the other hand, the single layer LSTM model performance closely trails the ARIMA model in its accuracy, yet the SVR on averages experience slightly greater accuracy having not undergone a log transformation.

| | **RMSE** | **sMAPE** |
|---|---|---|
| ARIMA (log) | 43.336 | 0.199 |
| ARIMA | 39.831 | 0.187 |
| SVR (log) | 11.405 | 0.0458 |
| SVR | 11.977 | 0.0504 |
| LSTM (log) | 45.161 | 0.209 |
| LSTM | 44.952 | 0.208 |

*Table 2: Accuracy Measures of Short-Term Forecast*

Viewing the development of the model the accuracy measures of the models have been plotted across all their individual forecasted intervals, yet Figure 17 shows the performance of the models across the rolling intervals where the jumps in performance are evident for ARIMA and LSTM models. This provides a view of the development of the accuracy across the 2023 which log-transformed and the un-transformed data is plotted separately, for comparability, but also for clarity. We have already seen very similar pattern of jumps in the Information Criteria of the identified ARIMA models. The accuracy measure in some instances for the ARIMA and LSTM model for certain periods of *t* experienced comparable results to the SVR model, dipping down to the same level. It is also clear that the magnitude of the change over time between models is lower with log transformed values.

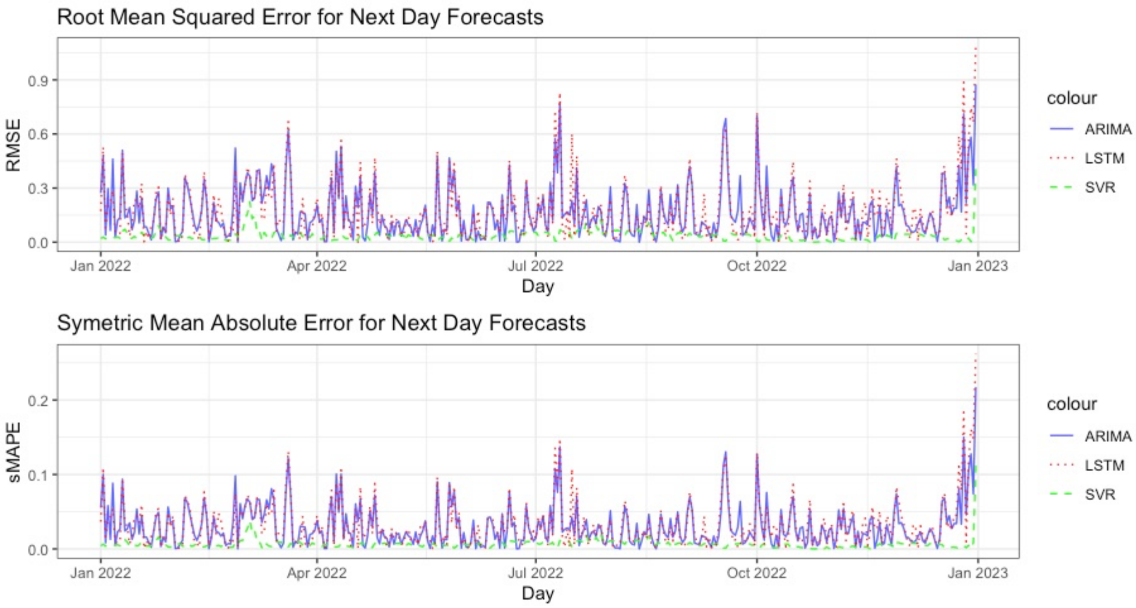


*Figure 26: Development of Log Transformed Short-Term Forecast Accuracy Across Rolling Intervals*

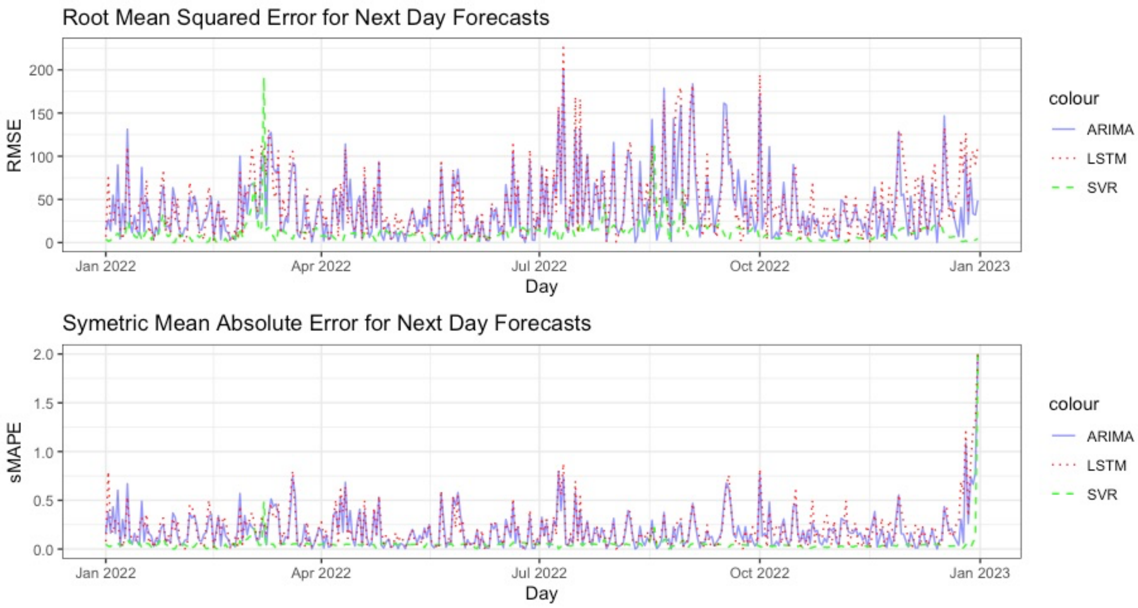


*Figure 27: Development of Short-Term Forecast Accuracy Across Rolling Intervals (Without transformation)*

## 3.4 Long Term Results

Once again, we may compare the three modelling approaches in terms of their ability to forecast based on the visual plotting across the individual rolling interval, their average accuracy scores, and the development of the accuracy measures, but now for the long term interval forecasting one month ahead .

We can see from the linked forecasted values against their test values with the vertical line indication of the separation of the multistep forecasts across all the intervals for each individual model across 2023 in Figures 28 to 23 below. It can be noted from the plot of the results the increased performance of the SVR model through the multistep forecast, although the slight increased differences between the forecast and the actual value in the third and eighth intervals. The ARIMA model struggles with applying multistep forecasts and produces a result that does not appear to closely follow the development of the data in both cases.

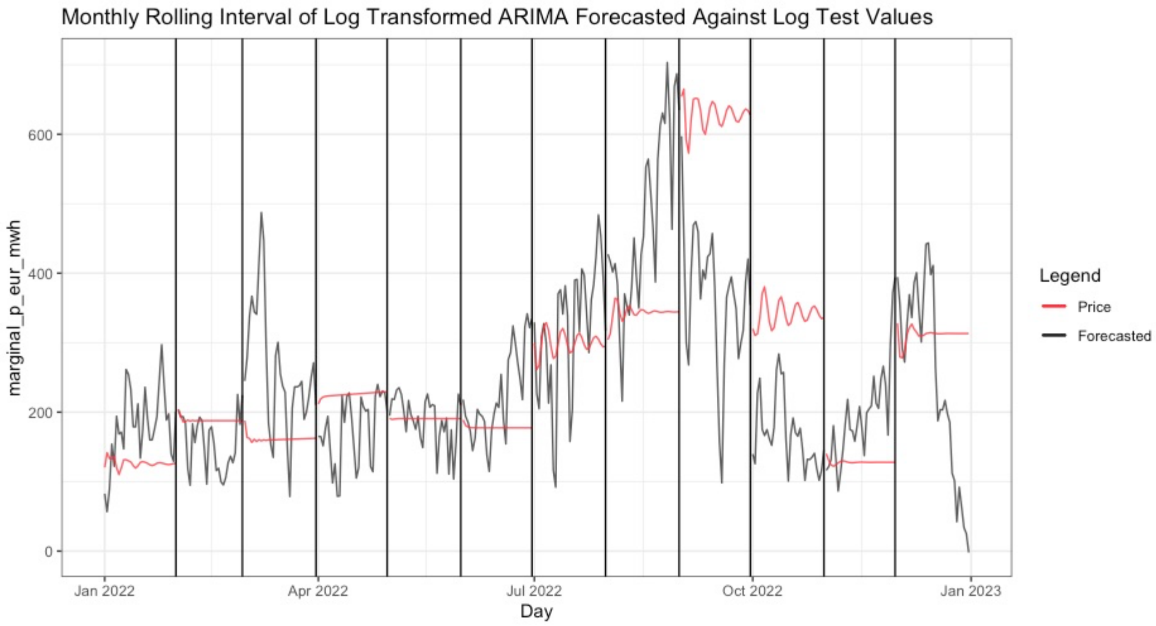


*Figure 28: ARIMA Multi Step Monthly Forecasted Price Against Test Values (Log Transformed)*

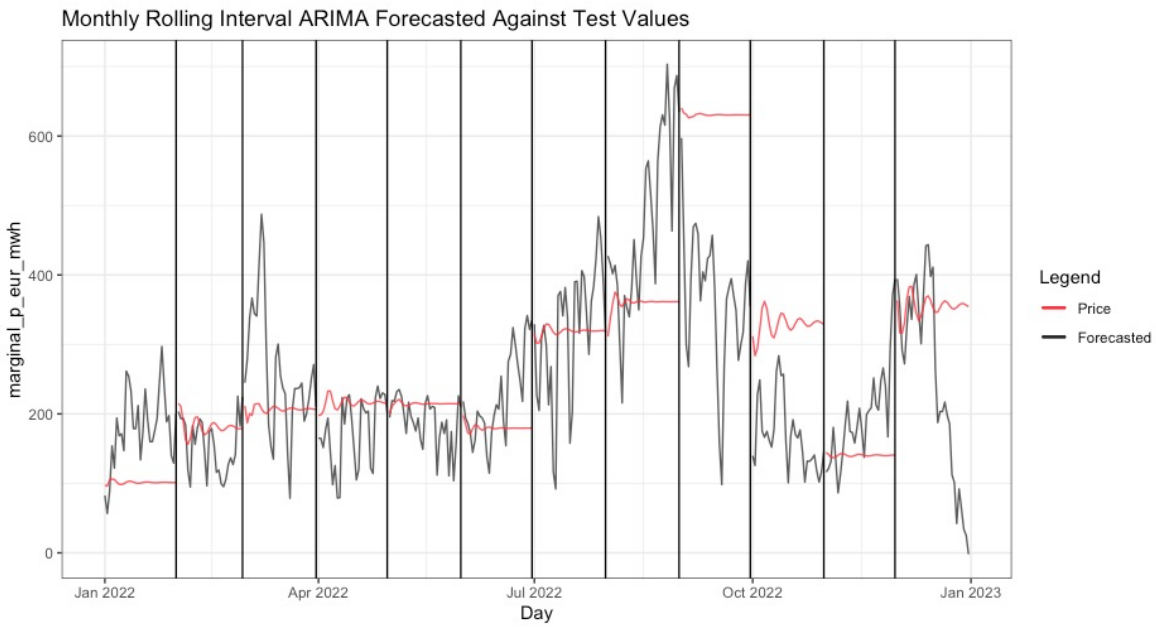


*Figure 29:ARIMA Multi Step Monthly Forecasted Price Against Test Values (Log Transformed*

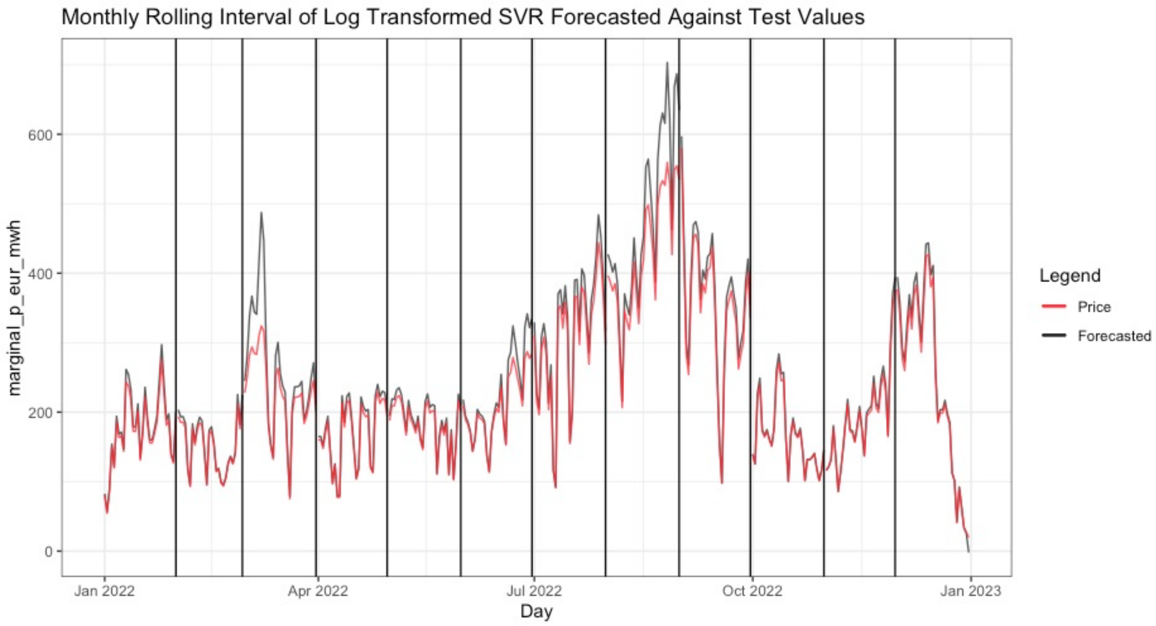


*Figure 30: SVR Multi Step Monthly Forecasted Price Against Test Values (Log Transformed)*

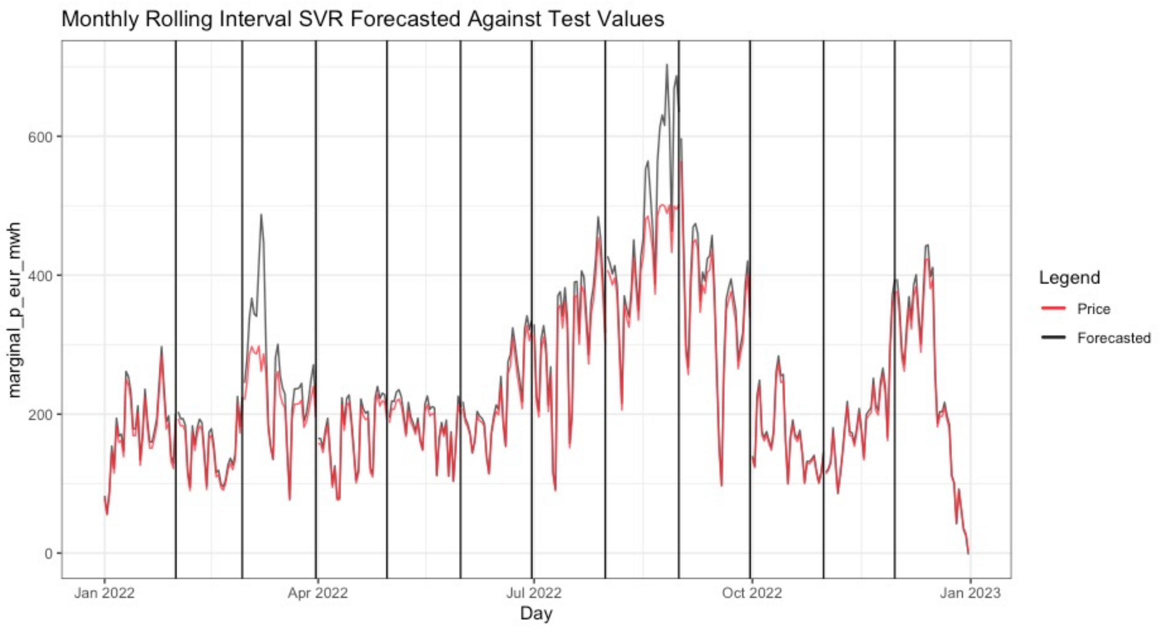


*Figure 31:SVR Multi Step Monthly Forecasted Price Against Test Values (Without Transformation)*

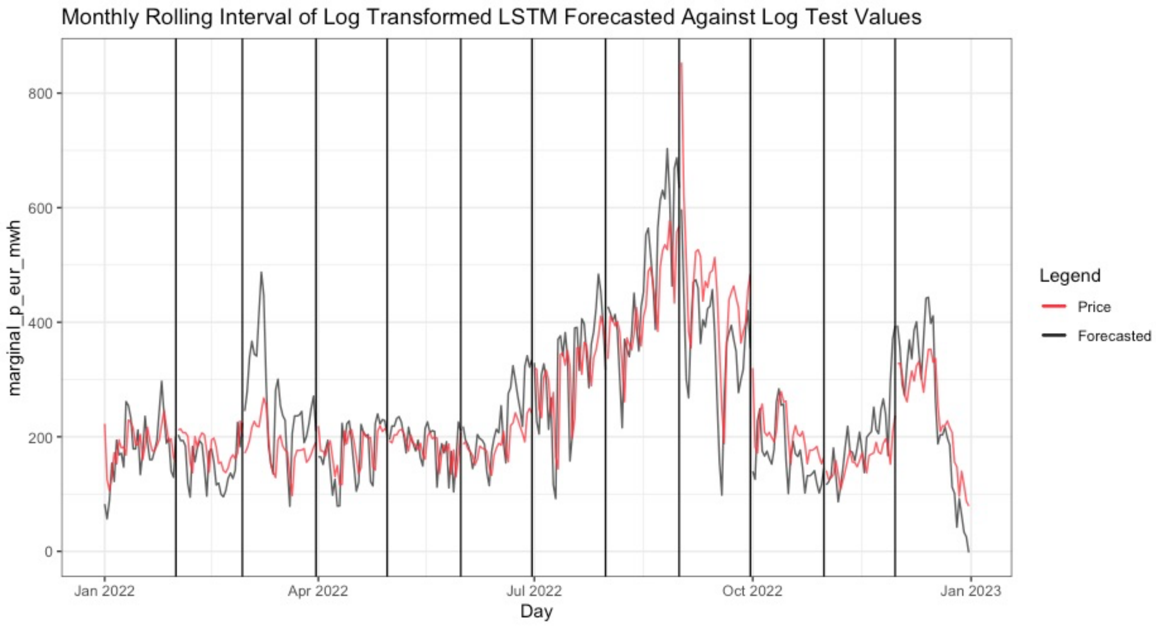


*Figure 32: LSTM Multi Step Monthly Forecasted Price Against Test Values (Log Transformed)*

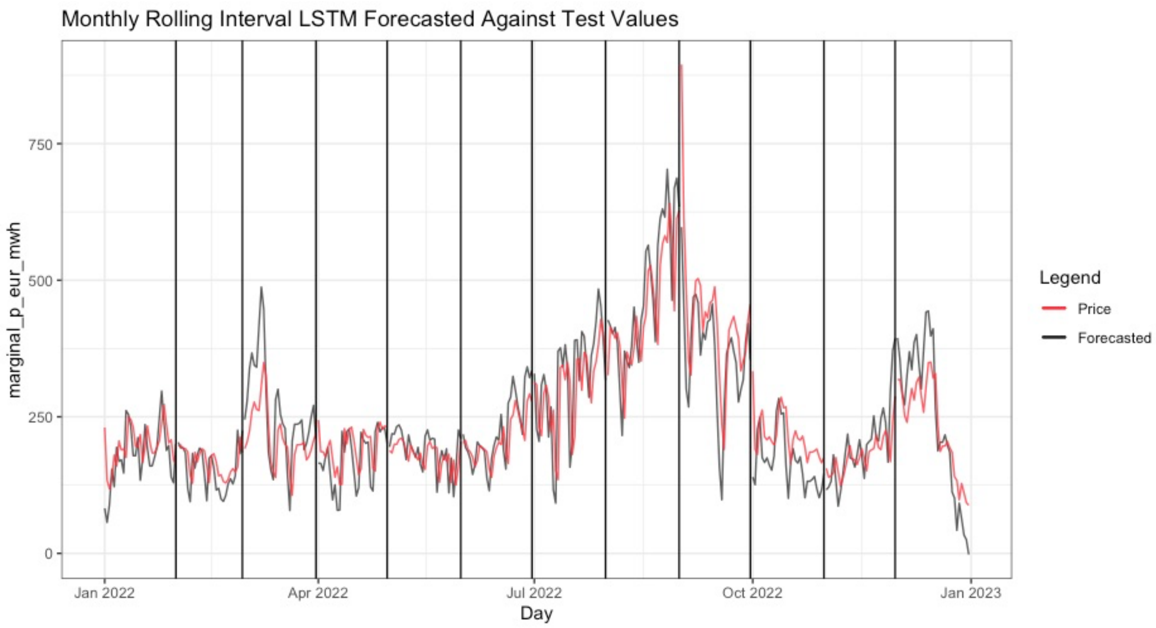


*Figure 33: LSTM Multi Step Monthly Forecasted Price Against Test Values (Without Transformation)*

The SVR model once again clearly performs the best in predicting the daily price across the monthly intervals accurately, where its mean RMSE and sMAPE, which we can see from the average accuracy measures in Table 3 is much lower in its value than both the ARIMA and LSTM process. On the other hand, the LSTM model outperforms ARIMA model in its accuracy to a greater magnitude than what was observed in the short-term forecast.

| | RMSE | sMAPE |
|---|---|---|
| ARIMA (log) | 118.198 | 0.386 |
| ARIMA | 116.585 | 0.373 |
| SVR (log) | 19.232 | 0.054 |
| SVR | 20.810 | 0.061 |
| LSTM (log) | 66.363 | 0.235 |
| LSTM | 61.363 | 0.219 |

*Table 3: Accuracy Measures of Long-Term Monthly Forecast*

To view the development of the model the accuracy measures of each models has been plotted across all their individual forecasted intervals. Figure 34 shows the performance of the models across the rolling intervals where the differences in the modelling performance is evident across all intervals. There is one occurrence in the eighth interval where the differences in the accuracy measures for the LSTM and SVR model slightly converge. Yet, we can see a few more cases where the ARIMA model, though performs worse, is closely comparable until the later stage intervals.

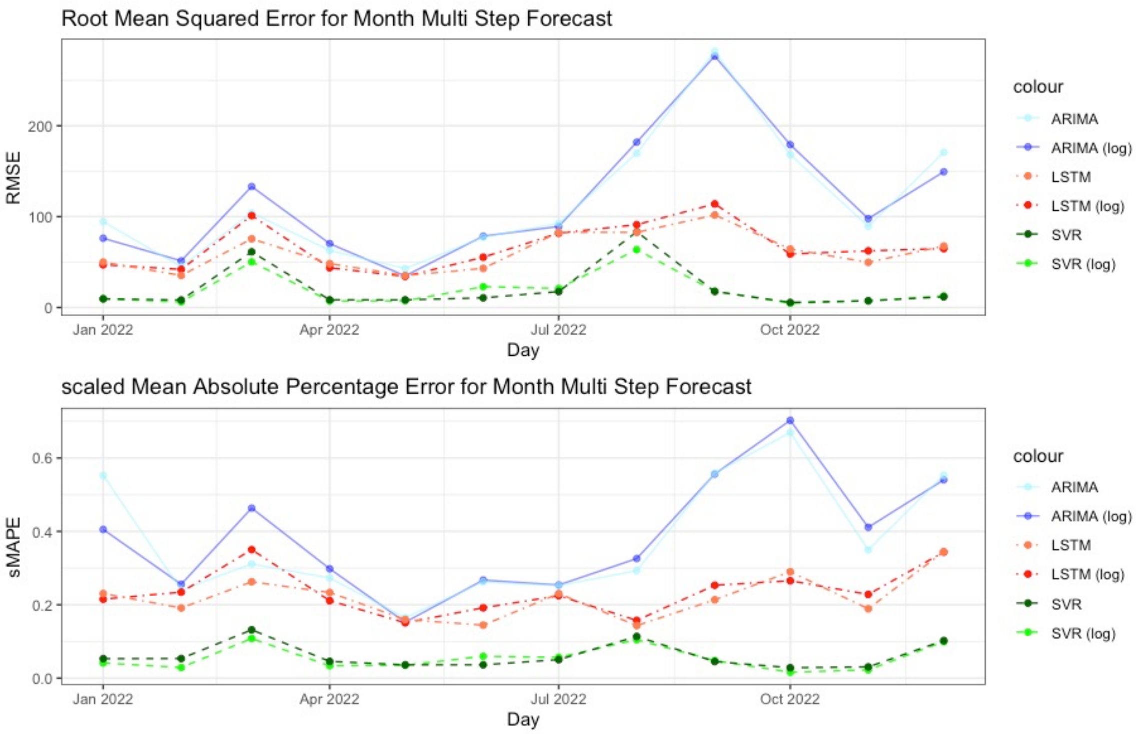


Figure 34: Development of Long-Term Forecast Accuracy Measure Across Rolling Intervals

# 4 Conclusion

This thesis aimed at investigating the effectiveness of various modelling in predicting electricity prices for the Czech Republic's day ahead. The practical part aimed at employing three different modelling techniques to predict a short-term horizon and a long-term horizon. The short-term horizon prediction being defined as a single step prediction of the next day's average prices, whereas the long-term prediction employed a MIMO multistep prediction over a monthly time horizon.

The dataset originated from the Czech OTE, the electricity market operator, where the data covered a four-year period between 2018 and 2022. The data set was split to provide a training set with a length of three years, with a one-year interval for the rolling interval approach to cover as a test set across long and short horizons. A simple variance reduction method was employed in the form of a log transformation, but throughout the study, non-transformed data was used in parallel.

The different modelling approaches employed in this thesis were ARIMA, SVR and LSTM neural networks. Predictions were made through a rolling interval with re-estimation of the model for each predicted interval rolling over the entire year of 2022. Performance of the predictions was measures by their average accuracy measure and the development of the accuracy measure over each interval.

In comparing the resulting models, the SVR model performed the best across both long- and short-term lengths in comparison with the other models yet had some differing performance between the transformed and un-transformed data. On average The ARIMA model outperformed the LSTM model in the short-term prediction, yet the results of the multistep prediction showed that the LSTM model was able to follow the pattern of the data better than the ARIMA model when it came to the long-term modelling approach.

While there are many possibilities to modelling electricity prices, as resulting from this study where the SVR model outperformed the other methods, for future studies of the topic of electricity price, there is room for further comparative analysis with the alternative approaches mentioned in Chapter 2.5. Expanding this study for multivariate modelling, would build off the topic broached in this study. The majority the literature is mainly focused on comparing Statistical methods with Computational Intelligence models, while alternatives models are not often included in comparison.

# List of References


1. Ahmed, N.K. *et al.* (2010) 'An Empirical Comparison of Machine Learning Models for Time Series Forecasting', *Econometric Reviews*, 29(5–6), pp. 594–621. Available at: https://doi.org/10.1080/07474938.2010.481556.

2. Allaire, J. and Chollet, F. (2023) 'keras: R interface to 'Keras'. Available at: https://CRAN.R-project.org/package=keras.

3. Allaire, J. and Tang, Y. (2024) 'tensorflow: R Interface to "TensorFlow"'. Available at: https://CRAN.R-project.org/package=tensorflow.

4. Awad, M. and Khanna, R. (2015) *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Springer Nature. Available at: https://doi.org/10.1007/978-1-4302-5990-9.

5. Bartz, E. *et al.* (2023) *Hyperparameter Tuning for Machine and Deep Learning with R: A Practical Guide*. Available at: https://link.springer.com/book/10.1007/978-981-19-5170-1 (Accessed: 4 June 2024).

6. Benecká, S. and Hošek, J. (2022) *What is driving the record-high growth in gas and electricity prices in Europe? - Czech National Bank*. Monetary Policy Report. Czech National Bank. Available at: https://www.cnb.cz/en/monetary-policy/monetary-policy-reports/boxes-and-articles/What-is-driving-the-record-high-growth-in-gas-and-electricity-prices-in-Europe/ (Accessed: 16 June 2024).

7. Benetton, M., Compiani, G. and Morse, A. (2023) *When cryptomining comes to town: High electricity use spillovers to the local economy*, *CEPR*. Available at: https://cepr.org/voxeu/columns/when-cryptomining-comes-town-high-electricity-use-spillovers-local-economy (Accessed: 13 November 2023).

8. Billinton, R., Chen, H. and Ghajar, R. (1996) 'Time-series models for reliability evaluation of power systems including wind energy', *Microelectronics Reliability*, 36(9), pp. 1253–1261.

9. Box, G.E.P. *et al.* (2015) *Time Series Analysis: Forecasting and Control*. Wiley & Sons, Inc (5th Edition). Available at: https://www.wiley.com/en-us/Time+Series+Analysis%3A+Forecasting+and+Control%2C+5th+Edition-p-9781118675021.

10. Box, G.E.P., Jenkins, G.M. and Reinsel, G.C. (2008) *Time Series Analysis*. Fourth. Wiley & Sons, Inc. Available at: https://onlinelibrary.wiley.com/doi/book/10.1002/9781118619193 (Accessed: 8 April 2024).

11. Carbaugh, B. and Sipic, T. (2017) 'Electric Utilities: How Electricity Is Priced', *The Journal of Energy and Development*, 43(1/2), pp. 193–211.

12. Cerqueira, V., Torgo, L. and Mozetič, I. (2020) 'Evaluating time series forecasting models: an empirical study on performance estimation methods', *Machine*

*Learning*, 109(11), pp. 1997–2028. Available at: https://doi.org/10.1007/s10994-020-05910-7.

13. Cherkassky, V. and Ma, Y. (2004) 'Practical selection of SVM parameters and noise estimation for SVM regression', *Neural Networks*, 17(1), pp. 113–126. Available at: https://doi.org/10.1016/S0893-6080(03)00169-2.

14. Colah, C. (2015) 'Understanding LSTM Networks', 27 August. Available at: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (Accessed: 21 April 2024).

15. Contreras, J. *et al.* (2003) 'ARIMA models to predict next-day electricity prices', *IEEE Transactions on Power Systems*, 18(3), pp. 1014–1020. Available at: https://doi.org/10.1109/TPWRS.2002.804943.

16. Cortes, C. and Vapnik, V. (1995) 'Support-vector networks', *Machine Learning*, 20(3), pp. 273–297. Available at: https://doi.org/10.1007/BF00994018.

17. Curran-Everett, D. (2018) 'Explorations in statistics: the log transformation', *Advances in Physiology Education*, 42(2), pp. 343–347. Available at: https://doi.org/10.1152/advan.00018.2018.

18. Dickey, D.A. and Fuller, W.A. (1979) 'Distribution of the Estimators for Autoregressive Time Series With a Unit Root', *Journal of the American Statistical Association*, 74(366), pp. 427–431. Available at: https://doi.org/10.2307/2286348.

19. Dubey, S.R., Singh, S.K. and Chaudhuri, B.B. (2022) 'Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark'. arXiv. Available at: https://doi.org/10.48550/arXiv.2109.14545.

20. Fildes, R. and Makridakis, S. (1995) *The Impact of Empirical Accuracy Studies on Time Series Analysis and Forecasting on JSTOR*. Available at: https://www.jstor.org/stable/1403481 (Accessed: 16 April 2024).

21. Gazzani, A. and Ferriani, F. (2022) *The impact of the war in Ukraine on energy prices: Consequences for firms' financial performance*, *CEPR*. Available at: https://cepr.org/voxeu/columns/impact-war-ukraine-energy-prices-consequences-firms-financial-performance (Accessed: 26 October 2023).

22. van Greunen, J. *et al.* (2014) 'The Prominence of Stationarity in Time Series Forecasting', *Studies in Economics and Econometrics*, 38(1), pp. 1–16. Available at: https://doi.org/10.1080/10800379.2014.12097260.

23. Gurney, K. (1997) *An Introduction to Neural Networks*. Available at: https://www.taylorfrancis.com/books/mono/10.1201/9781315273570/introduction-neural-networks-kevin-gurney (Accessed: 21 April 2024).

24. Hockreiter, S. and Schmidhuber, J. (1997) 'Long Short-term Memory', *Neural Computation* [Preprint]. Available at: https://doi.org/10.1162/neco.1997.9.8.1735.

25. Huisman, R., Huurman, C. and Mahieu, R. (2007) 'Hourly electricity prices in day-ahead markets', *Energy Economics*, 29(2), pp. 240–248. Available at: https://doi.org/10.1016/j.eneco.2006.08.005.

26. Hyndman, R. *et al.* (2024) 'forecast: Forecasting functions for time series and linear models'.

27. Hyndman, R.J. (2006) ‘Another look at forecast - accuracy metrics for intermittent demand’, *International Institute of Forecasters*, (4), pp. 43–46.

28. Hyndman, R.J. and Athanasopoulos, G. (2018) *Forecasting: principles and practice*. 3rd edn. OTexts. Available at: https://otexts.com/fpp3/.

29. Hyndman, R.J. and Khandakar, Y. (2008) ‘Automatic Time Series Forecasting: The forecast Package for R’, *Journal of Statistical Software*, 27, pp. 1–22. Available at: https://doi.org/10.18637/jss.v027.i03.

30. Ighravwe, D.E. and Mashao, D. (2020) ‘Analysis of support vector regression kernels for energy storage efficiency prediction’, *Energy Reports*, 6, pp. 634–639. Available at: https://doi.org/10.1016/j.egyr.2020.11.171.

31. James, G. *et al.* (2023) ‘Deep Learning’, in G. James et al. (eds) *An Introduction to Statistical Learning: with Applications in Python*. Cham: Springer International Publishing, pp. 399–467. Available at: https://doi.org/10.1007/978-3-031-38747-0_10.

32. Jędrzejewski, A. *et al.* (2022) ‘Electricity Price Forecasting: The Dawn of Machine Learning’, *IEEE Power and Energy Magazine*, 20(3), pp. 24–31. Available at: https://doi.org/10.1109/MPE.2022.3150809.

33. Kaňková, E. and Koľveková, G. (2020) ‘SPECIFIC FEATURES OF ELECTRICITY, ENERGY MARKET – PHENOMENON OF NEGATIVE PRICES’.

34. Keefe, B. (2022) *The Price of Energy Insecurity*. Available at: https://www.imf.org/en/Publications/fandd/issues/2022/12/POV-the-price-of-energy-insecurity-keefe (Accessed: 6 March 2024).

35. Kirchgässner, G., Wolters, J. and Hassler, U. (2012) *Introduction to Modern Time Series Analysis*. Springer Science & Business Media. Available at: https://link.springer.com/book/10.1007/978-3-642-33436-8.

36. Kunh and Max (2008) ‘Building Predictive Models in R Using the caret Package’. Journal of Statistical Software. Available at: https://www.jstatsoft.org/index.php/jss/article/view/v028i05.

37. Kwiatkowski, D. *et al.* (1992) ‘Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?’, *Journal of Econometrics*, 54(1), pp. 159–178. Available at: https://doi.org/10.1016/0304-4076(92)90104-Y.

38. Larsson, R., Clements, M.P. and Hendry, D.F. (2001) ‘Forecasting with difference-stationary and trend-stationary models’, *The Econometrics Journal*, 4(1), pp. S1–S19. Available at: https://doi.org/10.1111/1368-423X.00049.

39. Matalucci, S. (2021) *Climate change one cause of rally in energy prices, dw.com*. Available at: https://www.dw.com/en/energy-crisis-harsh-winter-would-add-fuel-to-climate-change-fire/a-59335095 (Accessed: 6 March 2024).

40. Needell, Z., Wei, W. and Trancik, J.E. (2023) ‘Strategies for beneficial electric vehicle charging to reduce peak electricity demand and store solar energy’, *Cell Reports Physical Science*, 4(3), p. 101287. Available at: https://doi.org/10.1016/j.xcrp.2023.101287.

41. OECD (2021) *Czech Republic 2021 Energy Policy Review*. Paris: Organisation for Economic Co-operation and Development. Available at: https://www.oecd-ilibrary.org/energy/czech-republic-2021-energy-policy-review_fd01b443-en (Accessed: 13 November 2023).

42. Ojemakinde, B.T. (2006) *Support Vector Regression for Non-Stationary Time Series*. University of Tennesee, Knoxville. Available at: https://trace.tennessee.edu/cgi/viewcontent.cgi?referer=&httpsredir=1&article=3107&context=utk_gradthes.

43. Peiqi, L. (2022) 'Time Series Forecasting Based on ARIMA and LSTM'. Available at: https://doi.org/10.2991/aebmr.k.220603.195.

44. Petnehézi, G. (2018) 'Recurrent Neural Networks for Time Series Forecasting'. arXiv. Available at: https://doi.org/10.48550/arXiv.1901.00069.

45. R Core Team (2024) 'A Language and Environment for Statistical Computing'. Vienna, Austria.: R Foundation for Statistical Computing, Vienna, Austria. Available at: https://www.R-project.org.

46. Schmidt, R.M. (2019) *Recurrent Neural Networks (RNNs): A gentle Introduction and Overview*, *arXiv.org*. Available at: https://arxiv.org/abs/1912.05911v1 (Accessed: 6 June 2024).

47. Scholkopf, B. and Smola, A. (2002) 'Support Vector Machines and Kernel Algorithms'. Available at: https://www.researchgate.net/publication/229025155_Support_Vector_Machines_and_Kernel_Algorithms.

48. Schwarz, G. (1978) 'Estimating the Dimension of a Model', *The Annals of Statistics*, 6(2), pp. 461–464.

49. Siami-Namini, S., Tavakoli, N. and Siami Namin, A. (2018) 'A Comparison of ARIMA and LSTM in Forecasting Time Series', in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Orlando, FL: IEEE, pp. 1394–1401. Available at: https://doi.org/10.1109/ICMLA.2018.00227.

50. Smola, A.J. and Schölkopf, B. (2004) 'A tutorial on support vector regression', *Statistics and Computing*, 14(3), pp. 199–222. Available at: https://doi.org/10.1023/B:STCO.0000035301.49549.88.

51. Stoft, S. (2011) *Power System Economics: Designing Markets for Electricity | IEEE eBooks | IEEE Xplore*. Available at: https://ieeexplore.ieee.org/book/5264048 (Accessed: 31 October 2023).

52. Svetunkov, I. and Petropoulos, F. (2017) 'Old dog, new tricks: a modelling view of simple moving averages'. Available at: https://doi.org/10.1080/00207543.2017.1380326.

53. Taieb, S.B. *et al.* (2011) 'A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition'. arXiv. Available at: https://doi.org/10.48550/arXiv.1108.3259.

54. Tashman, L. (2000) 'Out-of-sample tests of forecasting accuracy: An analysis and review'. Available at: https://www.researchgate.net/publication/223319987_Out-of-sample_tests_of_forecasting_accuracy_An_analysis_and_review (Accessed: 9 April 2024).

55. U.S. Energy Information Administration (2023) *Electricity explained - U.S. Energy Information Administration (EIA)*. Available at: https://www.eia.gov/energyexplained/electricity/ (Accessed: 12 November 2023).

56. U.S. Energy Information Administration (no date) *International - U.S. Energy Information Administration (EIA)*. Available at: https://www.eia.gov/international/rankings/country/CZE?pid=2&aid=4&f=A&y=01%2F01%2F2021&u=0&v=none&pa=88 (Accessed: 14 November 2023).

57. Weron, R. (2014) *Electricity price forecasting: A review of the state-of-the-art with a look into the future - ScienceDirect*. Available at: https://www.sciencedirect.com/science/article/pii/S0169207014001083 (Accessed: 6 April 2024).

# Appendices

GitHub Repository for the code used in this thesis:
https://github.com/christianbillinton/masters_thesis.git

Code 1 Function to create sets of rolling intervals (Short-term intervals)

```
# Function to create rolling window splits
rolling_interval_split <- function(data, train_window, test_window) {


  # Define window sizes in days
  train_days <- train_window
  test_days <- test_window

  # Initialize empty lists for train and test sets
  train_data <- list()
  test_data <- list()
  train_start = c()
  train_end = c()
  test_start = c()
  test_end = c()

  # Loop through the data with rolling window
  for (i in 1:365) {
    # Training data window
    train_start[i] <- i
    train_end[i] <- train_start[i] + train_days
    train_data[[i]] <- data[train_start[i]:train_end[i], ]

    # Test data window (1 day after training window)
    test_start[i] <- train_end[i] + 1
    test_end[i] <- test_start[i] + test_days
    test_data[[i]] <- data[test_start[i],]
  }

  return(list(train = train_data, test = test_data))
}
```

Code 2. Loop for Automatic ARIMA estimation, Prediction and Accuracy Measure Calculation (Short-term intervals)

```
#Model Training
```

```
fit_temp_stlog = c()
for(i in 1:365){
  fit_temp_stlog[[i]] = auto.arima(na.omit(split_day2$train[[i]]$log_marg_price))
}
#Prediction
fc_st_d = c()
f_st_val_d = c()
for(i in 1:sets_st){
  fc_st_d[[i]] = forecast(fit_temp_stlog[[i]], h = 1)
  f_st_val_d[[i]] = cbind(Day = as.POSIXct(split_day2$test[[i]]$Day),x =
as_tibble(fc_st_d[[i]]$mean))
}

#Accuracy Calculation
rmse_st_arimac = c()
smape_st_arimac = c()

for(i in 1:sets_st){
  rmse_st_arimac[i] = RMSE(f_st_val_d[[i]]$x, split_day2$test[[i]]$log_marg_price)
  smape_st_arimac[i] = smape(f_st_val_d[[i]]$x, split_day2$test[[i]]$log_marg_price)
  mean_st_arimac = c(RMSE = mean(rmse_st_arimac),
                     sMAPE = mean(smape_st_arimac))
}
```

Code 3. Loop for SVR Cross Validation and Training (Short-term intervals)

```
#Setting parameters for CV
SVRGridCoarse = expand.grid(.sigma=c(0.001, 0.01, 0.1), .C=c(seq(1,10, by = 1), 50, 100))
ctrl = trainControl(method = "cv", number=5)

#Loop for CV across Intervals
fit_temp_svr = c()
set.seed(123)
for(i in 1:sets_st){
  fit_temp_svr[[i]] = train(data = split_day2$train[[i]], log_marg_price ~
laglog_marg_price, method="svmRadial", tuneGrid=SVRGridCoarse, trControl=ctrl, type="eps-
svr")
}

set.seed(123)
for(i in 1:sets_st){
  fit_svr_st[[i]] = svm(log_marg_price ~ laglog_marg_price,
                            data = split_sv$train[[i]],
                            type="eps-regression",
                            kernel="radial",
                            cost= fit_temp_svr[[i]]$bestTune[2],
```

```
                            gamma = fit_temp_svr[[i]]$bestTune[1]
                            )
}
```

Code 4. SVR Cross Validation and Training (Short-term intervals)

```
#Setting parameters for CV
SVRGridCoarse = expand.grid(.sigma=c(0.001, 0.01, 0.1), .C=c(seq(1,10, by = 1), 50, 100))
ctrl = trainControl(method = "cv", number=5)

#Loop for CV across Intervals
fit_temp_svr = c()
set.seed(123)
for(i in 1:sets_st){
  fit_temp_svr[[i]] = train(data = split_day2$train[[i]], log_marg_price ~
laglog_marg_price, method="svmRadial", tuneGrid=SVRGridCoarse, trControl=ctrl, type="eps-
svr")
}

#Training
set.seed(123)
for(i in 1:sets_st){
  fit_svr_st[[i]] = svm(log_marg_price ~ laglog_marg_price,
                            data = split_sv$train[[i]],
                            type="eps-regression",
                            kernel="radial",
                            cost= fit_temp_svr[[i]]$bestTune[2],
                            gamma = fit_temp_svr[[i]]$bestTune[1]
                            )
}
```

Code 5.  Function and loop for LSTM (Short-term intervals)

```
lstm_pred = function(x_train, y_train, x_test, units_l1, epochs, b_size = 1){

  #initalize the sequential model
  model = keras_model_sequential()

  model %>% layer_lstm(units = units_l1,
                     batch_input_shape = c(1,1,1),
                     return_sequences = T,
                     stateful= F)%>%
    layer_dense(units = 1)

  model %>% compile(loss = 'mean_squared_error',
                  optimizer = "adam")
```

```
  model %>%  fit(
    x = x_train
    ,y = y_train
    ,batch_size = 1
    ,epochs = 10
    ,shuffle = FALSE)

  out = model %>%  predict(x_test, batch_size = b_size)

  return(out)

#keras session needs to be cleared from memory
  k_clear_session()
  rm(model)
}

#Loop with generated objects to deal with Keras “Modify in Place”
for(i in 1:sets_st){
  print(paste0("Run: ", i))

  assign(paste0("pred_st_lstm_",i), lstm_pred(x_train = lstm_train_st_laglog[[i]],
                                              y_train = lstm_train_st_log[[i]],
                                              x_test = lstm_test_st_laglog[[i]],
                                              units_l1 = 10,
                                              b_size = 1)
  )
  k_clear_session()

  #pred_val_st = rbind(pred_val_st, get(paste0("pred_st_lstm_",i))
}
```