2025

ECON4170 – Data science for Economists – Prediction model for electricity prices in Norway (NO1)

EXAM ASSINGMENT

Innhold

Intro

The task we wanted to tackle for this assignment was to find out if we could make a model to predict the electricity price in Norway (area NO1) one day ahead of time (24 hours). The assignment will include data gathering, data wrangling, creation of both lags and dummies, optimal lag selection and validation, sanity check and graph creation, model creation, model verification and optimization.

Data - gathering and wrangling.

The data that we gathered is mostly from ENTSO-E. These variables are electricity prices, reservoir levels and demand/load on electricity. ENTSO-E is a reliable source of data, and because of the transparency they advocate for, the data was very easily collected. The format of the electricity price was given in 15-minute increments, the reservoir levels we in weekly increments, and the load was hourly. To start with, we needed to make sure that all the data matched in terms of time increments. (we chose hourly increments, because this seemed the most rational). We needed to first change the time increments of both reservoir levels and electricity prices to hourly. The way we handled the electricity price increments was to take every 15-minute observation and find the meaning, then display the mean as the value for every hour instead. For the water reservoir levels, we needed to make the last observation carry forward for every hour, until the next observation (next week). So, the changes in the water levels are still weekly, but the value of that week is carried forward for every hour of that week. (ENTSO-E, 2025)

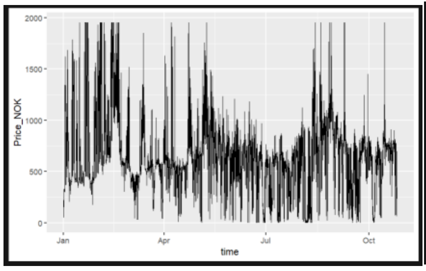
For the temperature we collected data from Norsk Klimaservicesenter. Since the data from ENTSO-E covered the data for area NO1 we had to collect data from 8 different weather stations in NO1 to give the same coverage. Those stations are "Oslo- Blindern, Gardemoen, fv319 Sagbukta, e134 Meheia, hamar- stavsberg, E6 vingnes, RV110 Skåra, Sarpsborg, E18 tveiten, Melsom". We take the average temperature across all stations and then have the AVGtemp_NO1. (Norsk Klimaservicesenter, 2025)

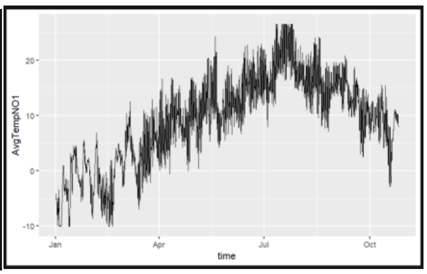
The data was then separately cleaned up and lined up and merged using time column. The start period was 01.01.2025 and end period is 25.10.2025. The separate data had the same solution for time daylight saving time in spring but, in autumn they did it differently instead of computing for that we cut the end before the time change and now the data was aligned.

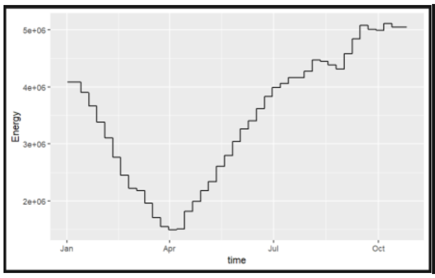
Model creation

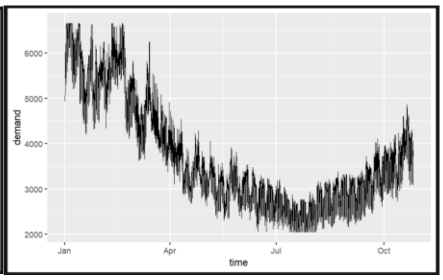
To get the model started, given that alle the variables are in the desired time format and time zone, with all NA values dropped, we started by joining alle the variables in the same dataset by using inner_join (). The start period was 01.01.2025 and end period is 25.10.2025. The separate data had the same solution for time daylight saving time in spring but, in autumn they did it differently instead of computing for that we cut the end before the time change and now the data was aligned. Now that everything is in the same dataset, we can remove any NA values that appear in the observations, by cutting off the dates on the variable with the least observation from the start of the year. (in our case, this was the electricity price, which did not have any observation after 27th of October). Furthermore, winsorizing the data set is important to cap out any sudden spikes in the data, such as price or demand. (Bobbitt, 2024)

Since the data is close to being ready for use, here is where you can start to look at the data in some graphs and do some correlations matrixes and sanity checks to make sure you have what you want for a good model. From the correlation matrix for price and temperature, we can see a decently strong negative correlation. Meaning a higher temperature meant a lower price. This is logical, given that places like Norway are usually colder, and therefore have a demand for heating when the temperature drops too far down. The correlation between the load of electricity and temperature is also a strong indication of this, with a very strong negative correlation that says if temperature goes up, the load goes down. The graphs also show some patterns, where the graph for demand and temperature is almost mirrored from each other, indicating the relationship between lower temperatures and higher load. (Datacamp, u.d.)









The next step in the model creation is to create some lags that will be used to give the model a better ability to predict outcomes and see patterns in the data. The lags included in this model are the price from yesterday (Price_lag_24), Price from last observation (one hour), price from a week ago, heating degree hours from yesterday, demand from yesterday and supply/reservoir levels from last week. In addition to the lags, we also added some dummies, this is mainly to see if there is any patterns in, for example, what time of day it is, what day it is in the week, what week of the year and month of the year. This makes for a more accurate model that can see the patterns and context for all the above-mentioned dates and times.

Optimal lag selection

We were supposed to select optimal lags but the method with AIC and BIC doesn't work unless the data is stationary and our electricity data wasn't supposed to be, so we switched to a more complex version. We were running a rolling cross validation with many different lags and the lags with lowest RMSE and MAE were chosen (Rob & George, 2025). We asked AI to create the code using this theory to find the optimal lag selection and validation for our dataset, since we could not find any way to do this in any of the course material, YouTube, data camp or stack overflow. The findings of the code were 1 hour, 2-hour, 24 hours, and one-week lags and that we remove the lags from the other variables. So, our initial guess of which lags to use were not so far off, but the addition of the 2-hour lag and removing of the other lag variables that are not the intercept is more optimal. (Tsay, 2010)

First model: Benchmark model

We want to make a model for the naïve benchmark. So that we have a baseline "goal" that we know we want to beat. The correlation between the current price and prices from one hour ago is very high, indicating that the price doesn't change much from hour to hour. This makes for some interesting data for predicting prices, but we are looking at prices 24 hours apart. The most effective way to make a benchmark is to just make a model of the different time formats we would want to look at. In this case, it is one-hour, one-day and one-week ahead of time and then take the RMSE of the models. This will give us a margin of error that our final model needs to outperform.

Naïve_1 benchmark: model assumes that the price, one hour ahead of time, is going to be the same value as the current price. This is to se if there are any trends in the change of prices when it comes to an hour-by-hour basis. It is very important for capturing short-term price changes. It is the simplest of the naïve tests.

Naïve_24 benchmark: model assumes that the price of tomorrow is the same as the current price, on the same hour. This helps to draw out if there are any strong daily cycles in the price. A practical example of this would be that more people use up more electricity at 15:00 rather than 03:00. (morning/ evening demand peaks)

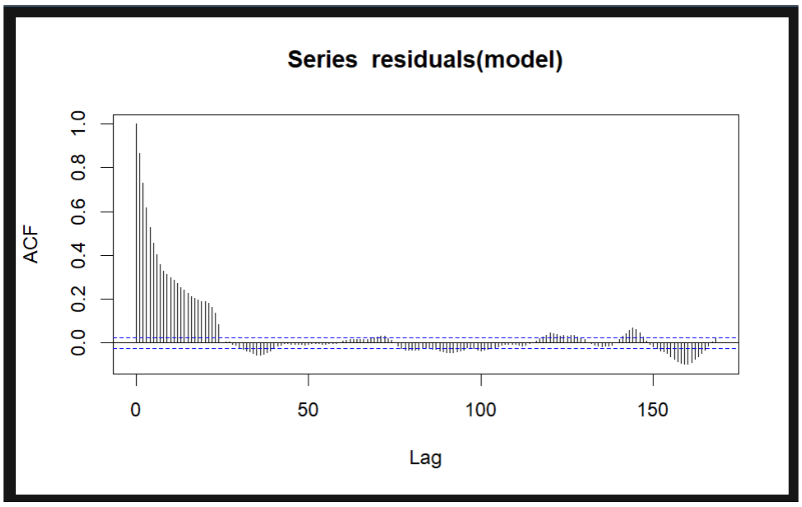
Naïve_168 benchmark: model assumes that the price of next week is the same as the current price. Helps to draw out patterns on a weekly basis. Such as weekday and weekend demand and supply. We also split the data into 80/20; this is to ensure a fair comparison when we make the final model later. 80% of the data is treated as the "training" part for the model, and the last 20% of the data is used for testing the model's ability to forecast. Furthermore, we test the accuracy of the benchmarks. This is done with the accuracy () command and gives us a compact set of values that we can use to see how accurate the benchmark is from the actual price.

From the metrics of the benchmarks, we can see that, as the correlation indicated, that the accuracy of the hour-to-hour prices is quite accurate, with an RMSE of 114. The daily RMSE was 266 and the RMSE of weekly naïve is 347. Intuitively this is logical, as the longer back you look, with the expectation of price being the same, the less accurate the model is.

Second model: ADL model

With the benchmark and goal for this assignment set out for us. We can start with the meat and bones of the project; finding a model that outperforms the benchmarks. The first model that we went for was a simple ADL model, which is the first interpretable price model that includes lagged prices, physical drivers, and seasonality. we get the final data set for the model, add the variable "y_next" which tells R-studios to make a target to price, 24-hours ahead of time, then drop any NA values caused by the lags, just to be sure before we do the regression.

Now we start building the model we want. The intercept is now the value we want to predict (y_next). We add different lags and variables that help the model with forecasting (lags and dummies). Fitting the OLS model with lm (), using the variables and data we want.



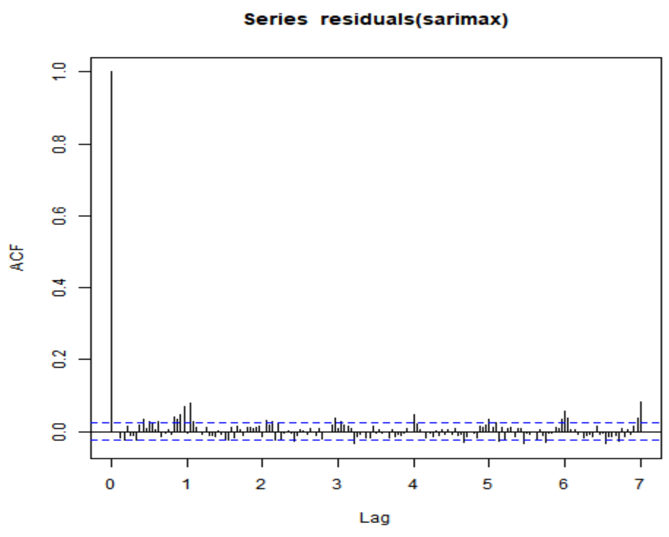
We wanted to do an ACF test and confirm our suspicion of autocorrelation. ACF test tells us how much the residual of a given dataset is autocorrelated. Too much autocorrelation means that the errors are too correlated with past values. In other words, our model might not be dynamic and doesn't fully find out how past prices effects future prices, prices are too dependent on their own recent values. For the ACF test, we found that our first model has a lot of autocorrelations, especially in the short lags (1-24 hour lags), this made us rethink if we wanted to use the ADL model, given that the autocorrelation in the ADL is not just "white noise".

The third model: SARIMAX

We concluded that the ADL model would not be the optimal choice, given the excessive amount of autocorrelation, we switched our focus to a SARIMAX model. The key difference in why we chose to switch to a more advanced model is that SARIMAX allows us to catch the effect of observations over time. ADL could not capture the residuals with strong autocorrelation, which we hoped the SARIMAX could do. This is the reason for why we chose to swap our model from ADL to SARIMAX. The reason why we chose ADL to start with was to have a more simple, smooth and easy model that was also consistent, but with the added ACF test, we concluded that a SARIMAX model approach would be the best way to move forward.

Making the SARIMAX model

The dataset is the same as we used in the ADL model. And the variables used are the same as well, except for the lags on the price itself, because the SARIMAX model already does this work for us. The Sarima model was relatively easy to build after already building the ADL model. It was just copy pasting previous code with a few adjustments and to make it fit the arima structure. However, when using the auto arima function it consistently gave us trouble when running the model and because it would try to force a d =1 or D = 1 differencing, which produced an error in the code. This happened because differencing made the model unstable and the optimizer returned nonfinite values. So, we experimented with the manual model and landed on our final model. Setting the specifications to (1,0,1)(1,0,1) period 24. This model still couldn't beat the ljung-box test meaning we still had autocorrelation (p-value under 0,05). This is because of no differencing but including it meant no model. We decided to move forward with this model even though it had autocorrelation as it was stable and working. Of course, there are some drawbacks as not all dynamics are captured, but the model is still acceptable for accurate forecasting, even with minimal amounts of autocorrelation.



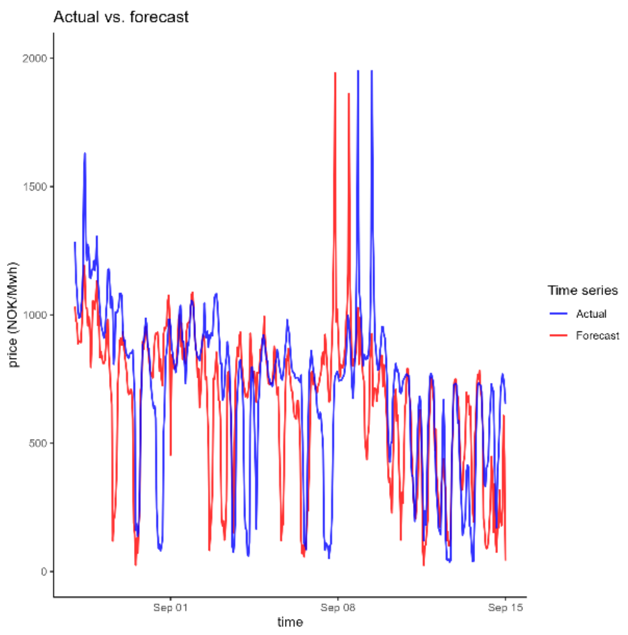
The summary of the model gives some key insight (SARIMAX.txt). SARIMAX uses one autoregressive term, and one moving average term for the model, with no differencing. It also uses one seasonal autoregressive term in daily cycles (24 hours, daily). The first three coefficients, AR1, MA and SAR1, are alle very important. AR1 = 0,876, this means that there is a very strong connection between the last price and the current price (one hour ago to now). MA1 = 0,154, this means that the price is mildly responsive to short-term fluctuations. SAR1 = -0.3636, this means a mild to moderate negative trend between the price from yesterday and the current price. HDH = 9.995, this means that for every degree colder than 18 degrees, the price can be expected to go up by about 9,995 kr. This strengthens the belief that lower temperatures lead to higher prices, which is logical, given that demand goes up as temperature drops. Demand (load) = 0,1232, this means that as demand goes up, the price goes up by 0,1232 kr. This is also logical, as this says that the more demand for electricity, the higher the price. This also fits well with the HDH findings. We also learn that energy (water reservoir levels) does not have any meaningful effect on prices, that's because its in weekly intervals, just stretched out over the whole week to fit the model's specifications. This left a bit of missing data/ better dataset to be desired.

The dummy variables are an interesting addition, we can see from the summary that the hourly dummies show a trend of electricity prices being the lowest at the earliest stages of the day (between hour 0-7), with negative coefs. The intuition would be that most people sleep, and most businesses are closed during these hours. Then we see an upward trend during the peak hours of the day. Indicating the opposite of the previous statement (everyone's awake and at work). Then, there was a slight drop during the afternoon and then a slight increase during the later hours of the day. The intuition here is that the weather usually drops as the sun sets, meaning more demand, meaning higher electricity prices. The average error of the model is RMSE = 115.6 and MAE = 71,4 (this is from an earlier outcast so it might not be as accurate as we hoped in the end but should serve the same purpose), this is not bad considering that our model, although advanced, lacks more information and variables that could make prediction more accurate. One concerning about the SARIMAX model is that the Ljung-Box test reveals that the p-value is very low, in a Ljung-Box test, if the p-value is below 0,05 that means that we can we can reject the null-hypothesis, and that indicates that there is still autocorrelation in our model (that it's not just white noise). (Geeksforgeeks , 2025)

Training the model (80/20 split in data)

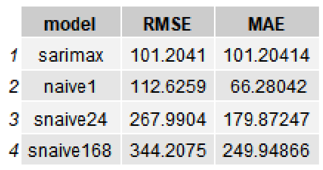
We did an 80/20 split on the data. 80% of the data was used for training the model and the last 20% was to test if our model could accurately predict the remaining 20% of the data. We used a rolling one-step-ahead forecast so that the model could not "look into the future" This will show how accurate the model is at predicting compared to the actual numbers from the dataset.

Comparing the models



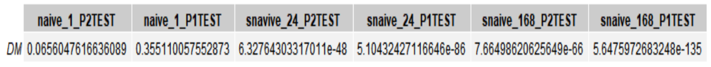
This shows the predicted vs actual forecast of the model. This graph is only a small period we split up to three graphs to better capture the difference between forecast and actual instead of it being cramped up. We see here that the forecast manages to capture the values, but the timing is off.

We saved the errors of the RMSE and MAE of the ARIMA model and compared the values to the benchmark models we had made earlier. We found out that our ARIMA model could predict the price of electricity with an RMSE of 101,20 (NOK/MWh) and an MAE of 101,2 (NOK/MWh). This value on the RMSE is the best and most accurate out of the models we set up, even beating out the one-hour benchmark test. And on the MAE value, only the one-hour benchmark test had a lower value than our model.



	model	RMSE	MAE
1	sarimax	101.2041	101.20414
2	naive1	112.6259	66.28042
3	snaive24	267.9904	179.87247
4	snaive168	344.2075	249.94866

This shows that our model outperforms the benchmark models, but we need to test whether the difference in forecast accuracy is statistically significant. So, we test using the Diebold-Mariano test, null hypothesis is both models have the same level of accuracy or alternative hypothesis whether the sarimax has lower forecast errors than the benchmark models.



	naive_1_P2TEST	naive_1_P1TEST	snavive_24_P2TEST	snaive_24_P1TEST	snaive_168_P2TEST	snaive_168_P1TEST
DM	0.0656047616636089	0.355110057552873	6.32764303317011e-48	5.10432427116646e-86	7.66498620625649e-66	5.6475972683248e-135

Here the p2 or p1 are power = 1 or power = 2. Power = 1 tests whether forecast errors are lower under MAE and power = 2 tests whether forecast errors are lower under RMSE. Here we test for both. Here as we can see we accept alternative hypothesis for every test except naïve 1 model. Meaning our model does not have a significantly higher level of accuracy than the naïve 1 model. But our model is significantly better than both the naïve 24 and 168. (Tsay, 2010) (Hardy, 2025) (Triacca)

Testing different models

We also wanted to test what contributes to better forecasting performance, so we stripped the model and made a simple model with only dummy hours and HDH and demand. We found that RMSE 114 ,8 and MAE 70,57 for the simple model is less than RMSE and MAE for our actual model which includes dummy month, dummy weekday and Energy. We already know from the sarimax summary that Energy doesn't contribute but now we know that dummy weekday and dummy month also don't increase forecast performance. We earlier thought that including month and weekday helps the model distinguish between colder and warmer seasons. But all that truly matters for the model is the demand and HDH. As this captures the differences in the months because if its January the HDH would be low and demand is high the model understands that and doesn't need the dummy month and weekday.

Conclusion

Our model managed to beat both the naïve 24 model and naïve 168 model. The goal was to be able to predict electricity prices 1 day ahead and to test the model up against the naïve 24 benchmark. Our model outperformed the benchmark, and it had statistically significantly lower forecast errors than the benchmark model. We also better understand how the model works and what variables increase performance. We earlier thought adding dummies for month and weekday would increase forecasting performance, but we have learned that's not the case as HDH, and demand covers that.

How we used AI

We used ai mainly for the optimal lag selection and rolling forecast. In the r. script we have marked which code was created by AI. Other than that, it was used to help transform the ADL model into the Arima model. But here the code was mainly already written just needed to be altered. For the rolling forecast it was mostly coding self but with help of ai to understand how it was done. But for the optimal lag selection the whole chunk was ai generated. The code was generated by using theory and step by step translating the theory into code using AI. In data camp we also use ai explanation tool in the tasks to understand some functions and code we used.

Referanser

Bobbitt, Z. (2024, Mai 27). *statology.org*. Retrieved from stsatology: https://www.statology.org/r-winsorize/

Datacamp. (n.d.). *datacamp.com*. Retrieved from https://campus.datacamp.com/courses/introduction-to-the-tidyverse/data-visualization?ex=1

ENTSO-E. (2025). *https://transparency.entsoe.eu/market/energyPrices?appState=%7B%22sa%22%3A%5B%22BZN%7C10YNO-1--------2%22%5D%2C%22st%22%3A%22BZN%22%2C%22mm%22%3Atrue%2C%22ma%22%3Afalse%2C%22sp%22%3A%22HALF%22%2C%22dt%22%3A%22TABLE%22%2C%22df%22%3A%222025-10-25%22%2C%22tz*. ENTSO-E.

Geeksforgeeks . (2025, Juli 23). *geeksforgeeks.org*. Retrieved from geeksforgeeks: https://www.geeksforgeeks.org/python/complete-guide-to-sarimax-in-python/

Hardy, R. (2025, november 23). *stats.stackexchange*. Retrieved from stats.stackexchange: https://stats.stackexchange.com/questions/139462/diebold-mariano-test-for-predictive-accuracy

Hyndman, R. J., & Anthanasopoulos, G. (2018). *Forecasting: principles and practice*. Melbourne.

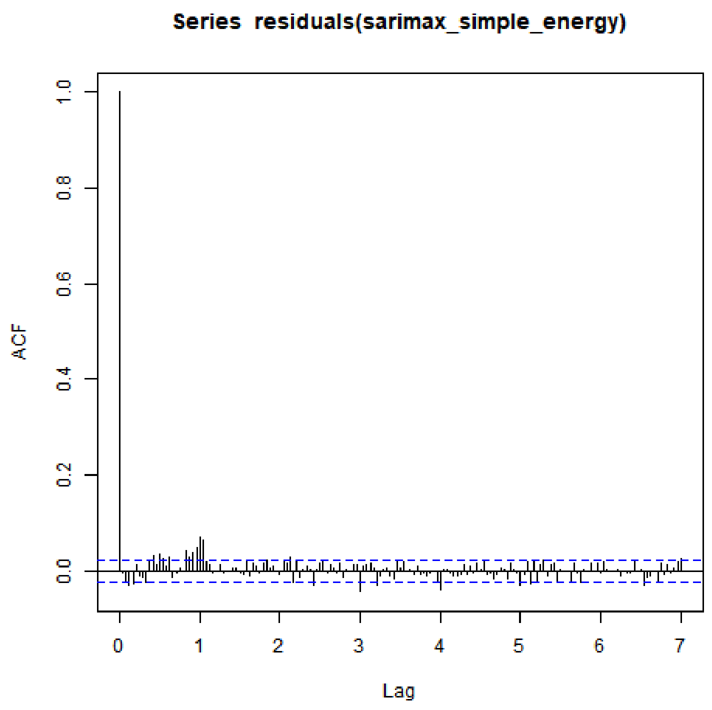
Norsk Klimaservicesenter. (2025, november 23). *seklima.met.no*. Retrieved from seklima.met.no: https://seklima.met.no/observations/%20for%20lufttemperatur%20i%20NO1.%20Oslo-

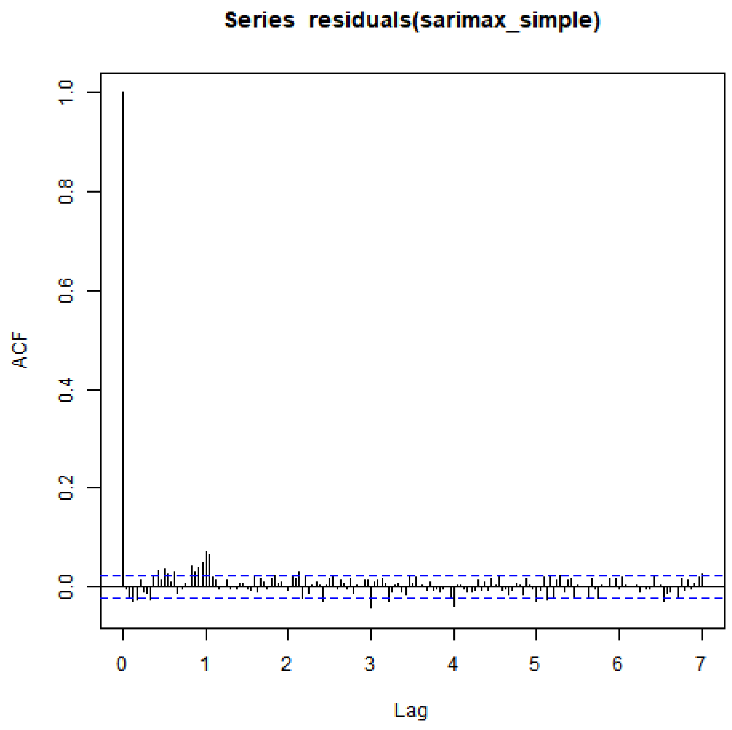
%20Blindern,%20Gardemoen,%20fv319%20Sagbukta,%20e134%20Meheia,%20hamar-%20stavsberg,%20E6%20vingnes,%20RV110%20Sk%C3%A5ra,%20Sarpsborg,%20E18%20tveiten,%20Melsom.

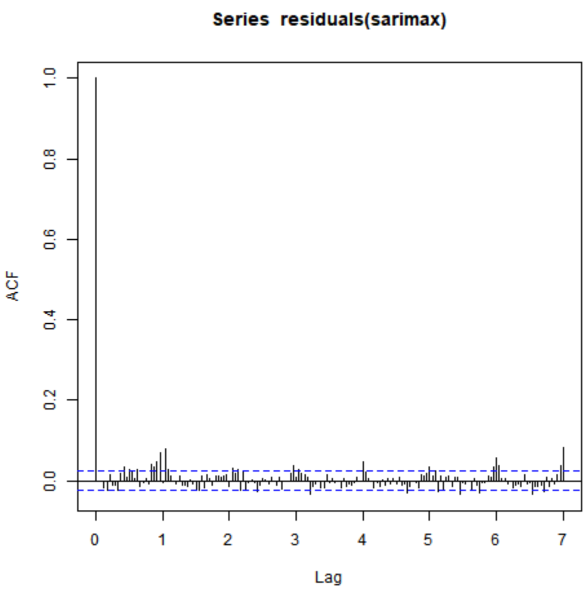
Triacca, U. (n.d.). *Lesson19: Comparing Predictive.*

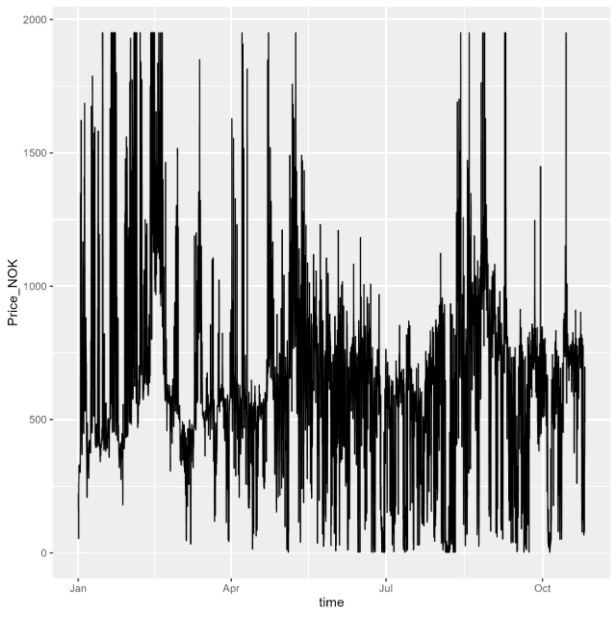
Tsay, R. S. (2010). *Analysis of financial time series.* New Jersey: John Wiley & Sons.

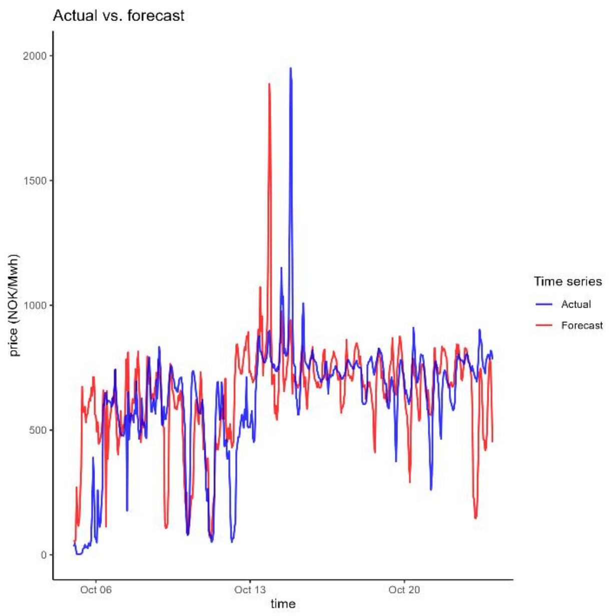
Appendix

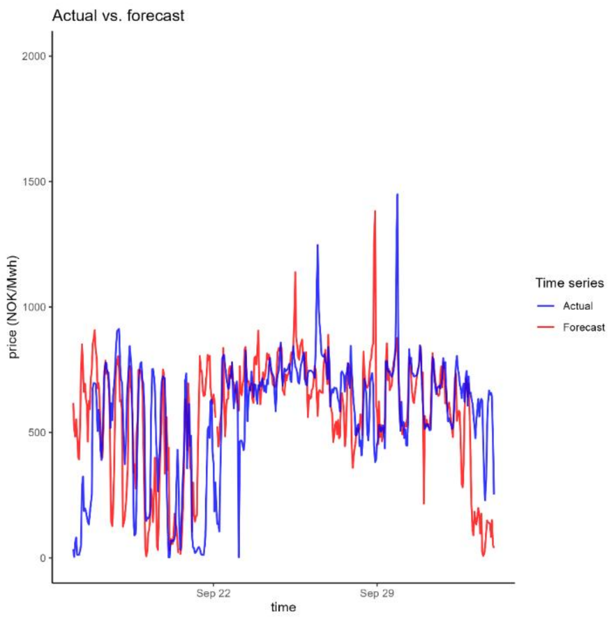

Series residuals(sarimax_simple_energy)
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
7
Lag

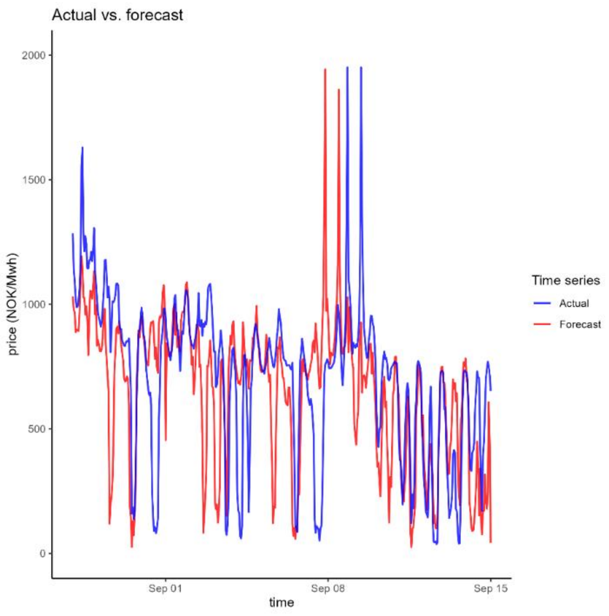

Series residuals(sarimax_simple)
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
7
Lag

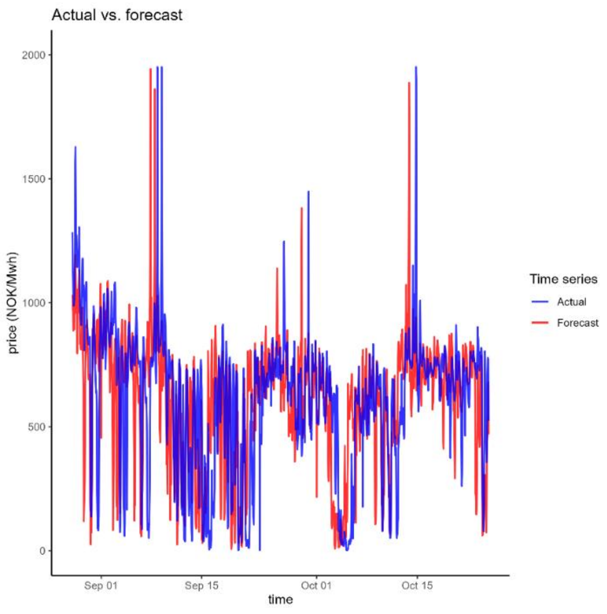

Series residuals(sarimax)
ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6
7

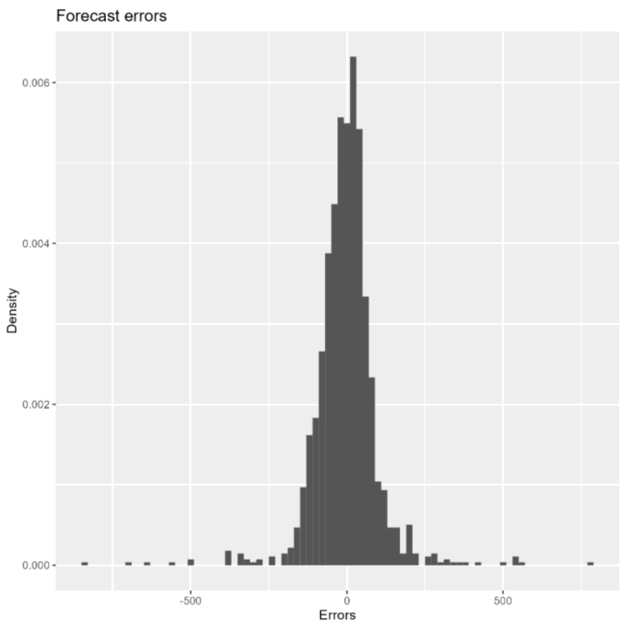

Price_NOK
time
0
500
1000
1500
2000
Jan
Apr
Jul
Oct

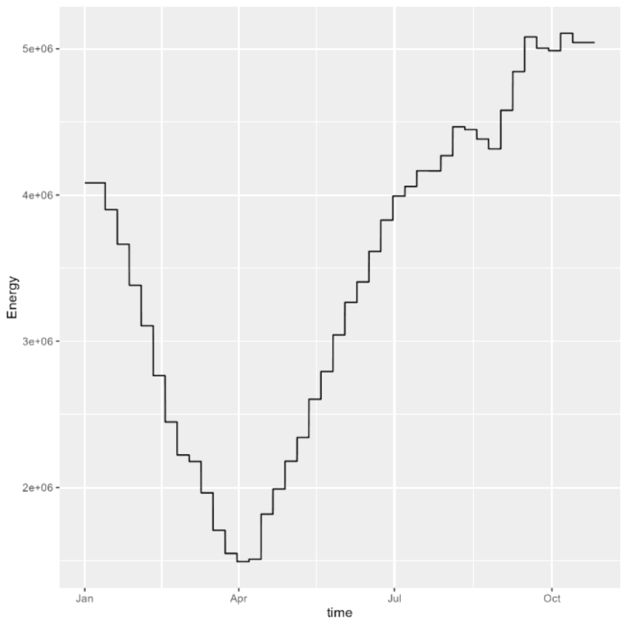

Actual vs. forecast
price (NOK/Mwh)
time
Time series
Actual
Forecast
0
500
1000
1500
2000
Oct 06
Oct 13
Oct 20

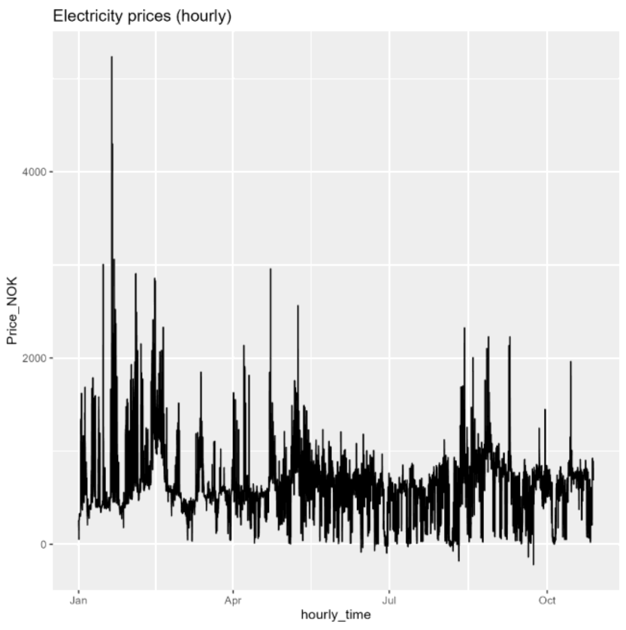

Actual vs. forecast
price (NOK/Mwh)
2000
1500
1000
500
0
Sep 22
Sep 29
time
Time series
Actual
Forecast

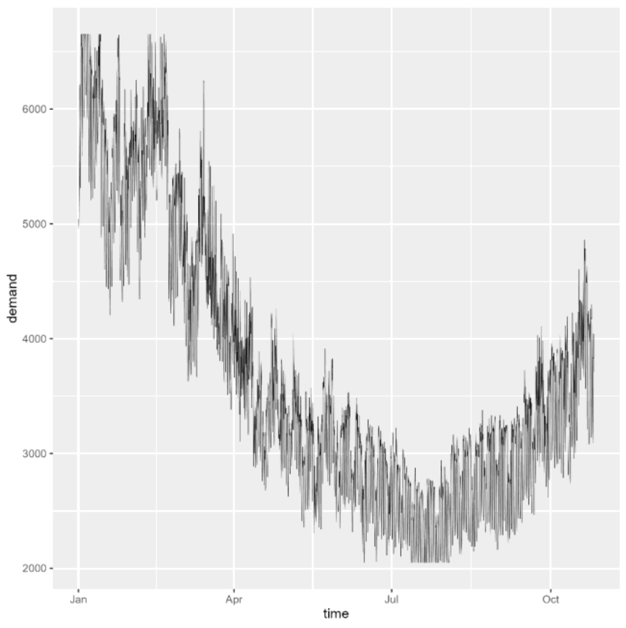

Actual vs. forecast
price (NOK/Mwh)
2000
1500
1000
500
0
Sep 01
Sep 08
Sep 15
time
Time series
Actual
Forecast

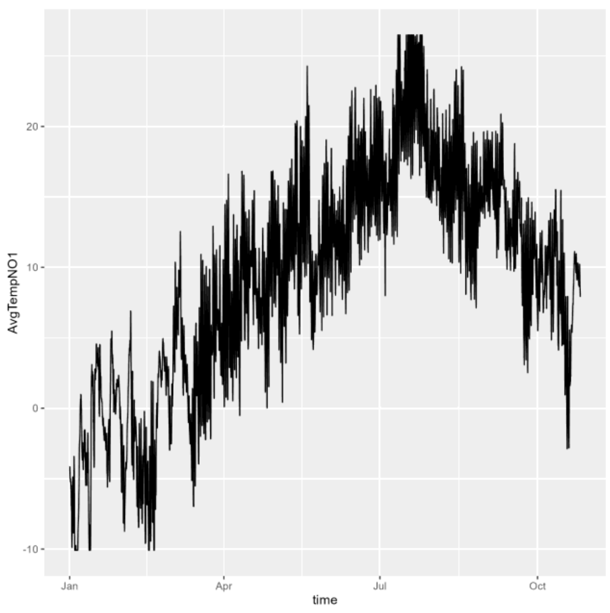

Actual vs. forecast
price (NOK/Mwh)
2000
1500
1000
500
0
Sep 01
Sep 15
Oct 01
Oct 15
time
Time series
Actual
Forecast

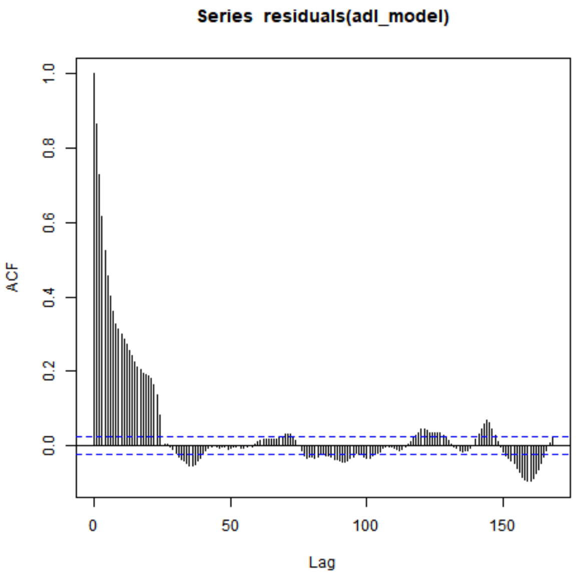

Forecast errors
Density
0.006
0.004
0.002
0.000
-500
0
500
Errors

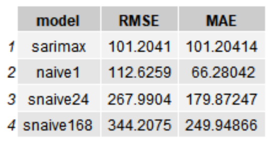

Energy
5e+06
4e+06
3e+06
2e+06
Jan
Apr
Jul
Oct
time


Electricity prices (hourly)
Price_NOK
4000
2000
0
Jan
Apr
Jul
Oct
hourly_time

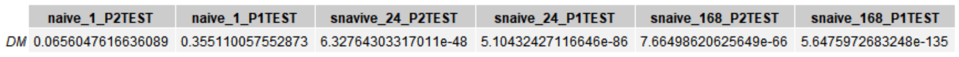

demand
6000
5000
4000
3000
2000
Jan
Apr
Jul
Oct
time







	model	RMSE	MAE
1	sarimax	101.2041	101.20414
2	naive1	112.6259	66.28042
3	snaive24	267.9904	179.87247
4	snaive168	344.2075	249.94866



	naive_1_P2TEST	naive_1_P1TEST	snavive_24_P2TEST	snaive_24_P1TEST	snaive_168_P2TEST	snaive_168_P1TEST
DM	0.0656047616636089	0.355110057552873	6.32764303317011e-48	5.10432427116646e-86	7.66498620625649e-66	5.6475972683248e-135