UNIL | Université de Lausanne

HEC Lausanne

Project I

STUDY OF THE DOMOHOURLY DATA

How can we best forecast the meteorological measures of an experimental house in Spain in the DomoHourly dataset?

Forecasting II

Spring 2019

Group C

HEC Lausanne

Submitted on

May 22, 2019

DECLARATION OF AUTHORSHIP

This project was written by us and in our own words, except for quotations from published and unpublished sources, which are clearly indicated and acknowledged as such. We are conscious that the incorporation of material from other works or a paraphrase of such material without acknowledgement will be treated as plagiarism, subject to the custom and usage of the subject, according to the University Regulations. The source of any picture, map or other illustration is also indicated, as is the source, published or unpublished, of any material not resulting from our own research.

Rita Sefraoui Tahiri (10812097)

Charlotte Fleury (14608749)

Alexandre Schroeter (15400716)

Clément Perez (14406482)

SUMMARY

In the context of the forecasting class, we want to forecast five different meteorological measures (humidity outside, humidity inside the dining room, CO2 inside the dining room, lighting inside the dormitory, and temperature inside the dining room) of an experimental house in Spain. We want to obtain 1-hour, 24-hour and 5-day forecasts (120 hours). We base it on a database of the real values for these measures for 12 days in April 2012. We first perform an exploratory analysis of each variable, plotting it through time, and trying to see if there are any seasonal patterns or trends. We obtain that humidity in the dining room (humedad comedor sensor) displays an increasing trend with a weak seasonal pattern. Humidity outside (humedad exterior sensor) has a stronger seasonal pattern with also an increasing trend. Temperature comedor sensor has a very strong seasonal pattern with a trend that seems to go first upwards and then slightly downwards. CO2 comedor sensor (CO2 inside the dining room) displays a strong horizontal trend with what looks like random peaks with a very weak seasonality. Finally, lighting in the dormitory (lighting habitacion sensor) displays a very strong seasonal pattern with very little variation to it.

Given these first results and that all variables display weak or strong daily seasonality, we test forecasting methods adapted to seasonal patterns: Holt-Winters' for the two humidity measures, ETS for all the variables (including the two humidity measures), and ARIMA for all variables.

For each method tested, we look at the AIC of each variable and looked for the smallest ones. We also compare the residuals given by each method to select the ones that display the most stationary, random patterns with uncorrelated ACF and a normal distribution. We want the number of lags outside of the ACF limits to be less than 5% to reduce autocorrelation of the errors. In our case, these two methods always give us the same model to select. Indeed, the one with the smallest AIC was also consistently the one with the best uncorrelated stationary residuals. For humedad comedor sensor, we got an ARIMA(2,1,1)(1,0,1)[24], confirming a weak seasonality. For the humedad exterior sensor we select an ARIMA(2,0,2)(1,1,0)[24]. For the CO2 comedor sensor, we use an ARIMA(1,1,2)(2,0,0)[24]. Temperature comedor sensor was best forecasted with an ARIMA(2,1,0)(2,1,0)[24], confirming a very strong seasonal daily pattern. Finally, lighting habitacion sensor also confirms the strong seasonal daily pattern with an ARIMA(1,0,0)(1,1,2)[24]. Once we obtain these results, we verify the stationarity of the errors with a Kwiatkowski-Phillips-Schmidt-Shin test and an augmented Dickey–Fuller test. Both confirm each time that our errors were indeed stationary. We verify the invertibility of our methods before performing the 1 hour, 24 hours and 120 hours forecast. Finally, to evaluate the forecasts we perform a cross-validation to calculate the RMSE of each method. The best forecasting method is the one for lighting habitacion sensor, with the lowest RMSE for each forecast. The biggest RMSE are for the most random variables : CO2 comedor sensor and humedad exterior sensor.

TABLE OF CONTENTS

INTRODUCTION

In the context of the forecasting course in the Master of Management of the University of Lausanne, we analyze a database with measurements of meteorological data for an experimental house in Spain. These measurements are collected each hour during 12 consecutive days, from April 18 to April 30. The variables measured were the temperature (outside, in the « comedor » which is the dining room, in the « habitacion » which is a dormitory), the CO2 concentration (in the comedor and in the habitacion), the humidity (in the comedor and habitacion), the lighting (in the comedor and habitacion), precipitations (outside), various measures of wind, pressure, and sun exposure. The aim is to be able to predict the values one hour, one day and five days in advance for the temperature in the dining room, the CO2 in the dining room, the humidity in the dining room, the lighting in the room and humidity outside. Can we effectively forecast the measures through time of temperature, CO2, and humidity in the living room, lighting in the dormitory and humidity outside up to 5 days ? And what are the best methods for each of these variables ?
For each variable, we first perform an exploratory analysis to see if there are any relevant patterns in the measures. This helps us narrow down the methods to test for the forecasting part. After that, we test several methods of forecasting (like ARIMA and Holt-Winters) using RStudio, and select the methods that seem the most appropriate by testing them and comparing them. In this report in particular, we choose to compare the residuals of each method and select the methods having the most stationary, random (with no pattern), normally distributed and uncorrelated residuals. Then we apply the method and forecast the 1-hour, 1-day and 5-day intervals. Finally, performing a time-series cross-validation, we evaluate the quality of our forecasts with RMSE measures.

DATA DESCRIPTION

We receive a .RData file of hourly mereological measures from April 18 to April 30, 2012. Having a look at the whole database, we see some columns (precisely Exterior_entalpic1, 2 and turbo) equals to zero all the time. We eliminate them from the analysis as they do not provide us any information. We also look at overall correlations between variables to see if there is any obvious link and if any measures are redundant. It seems that all temperatures are extremely correlated (Figure 1: Correlation of temperature measurements). CO2 of comedor and habitacion are very correlated too (88,1%), and so are measures of humidity in these two rooms (96,2%) (

Figure 2: Correlations of CO2 measures and humidities). Lightings are correlated between them and so are the measures labelled “meteo”. When considering these computations, we could say that predicting only some of the variables will give us a good insight into the others. Concerning our five variables of interest, humedad exterior sensor and humedar comedor sensor are the most correlated (77,7%) whereas are all others are under 50% correlation (Figure 3: Correlation between our variables of interest). This means that these measures are in general quite unrelated and can gives us a good insight into what is going to happen in this house for the next hour to five days.

We create a dataframe in R containing our five variables of interest like stated in the introduction. This data frame consists of the “hour”, “day”, “month” and “year” along with “Temperature comedor sensor”, “CO2 comedor sensor”, “Humedad comedor sensor”,” Lighting habitacion sensor”, and “Humedad exterior sensor”. “Temperature comedor sensor” is a *numerical* measure in *degree Celsius* with no missing value. “CO2 comedor sensor” is a *numerical* measure of *particle per million* with no missing value either. “Humedad comedor sensor” is a *numerical* percentage value with no missing measure.

"Lighting habitacion" sensor is a *numerical* value in *lux* with no missing value, and finally, "Humedad exterior sensor" is also a *numerical* relative percentage of humidity and all values are present. Apart from isolating the columns we need and putting them in a R dataframe format, no further general transformation are performed.

METHODS

For each variable of interest, we first perform an exploratory analysis, test different methods, select a method, apply it and then evaluate it.

- For the exploratory analysis, we use the "ggplot" package of R to visualize the data and its behavior. First at a macro level, we use autoplot() to see if there is a seasonality pattern, a trend or another notable pattern. In order to investigate the seasonality of each variable, we use the method ggseasonplot() which allows us to see the evolution of each measure per hour, each day in a different color. For example, we would see April 18 in red, day April 19 in blue and so on, in the abscissa axis we would have the hours of the day from 0 to 24. It allows us to see if there is any pattern that repeats each day. For the same purpose, we use ggsubseriesplot() method, it shows the mean and variance of the measures for each hour throughout the days of the month.
  We use a STL decomposition from the package "stats" to decompose the variable into a trend, a seasonality, and a remainder (after subtracting the seasonality and trend to the real data measurement). It allows us to have a better insight into which methods may or may not be necessary to test for the forecast. We use a STL method instead of an additive one because the additive method does not give us robust decomposition if there is a varying variance. We use a STL method instead of a multiplicative one because the remainders with the STL decomposition are smaller and show less seasonality than with the multiplicative method. As the purpose of this report is to analyze the forecasts of the variables and not so much the decomposition, we decided not to include the other decompositions in the results. If the variance observed during the autoplot is not stable, a BoxCox transformation (with the package "forecast") of the data is used before introducing the data into the forecasting methods, unless we used auto.arima. Indeed, the function performs the best fitting BoxCox transformation on its own if stipulated in the arguments, which we do. Any BoxCox transformation is explicitly stated for each variable in the results. In the same way, if there is a need for more stationarity, a differentiation (diff) method is used and explicitly stated each time.
- The forecasting methods used and tested are arima and auto.arima from the "forecast" package, which tests which arima model would better fit. We all allow for a drift in the arguments and use a false stepwise comparison of model for the R function to return the best model according to the AICc criterion. We use a lambda=" auto" to perform automatically the optimal BoxCox transformation for variance stabilization and biasadj=TRUE to transform back the forecasting results. For some variables, we also test Holt-Winters models from the "ses" package which uses exponential smoothing with seasonality to compute a forecast. We finally test ETS function from the same package which is also based on exponential smoothing.
- To choose which method to apply for the forecasting, we apply checkresiduals() to the methods from the forecasting package which gives us the distribution of the residuals (to see if they are normally distributed) , the ACF of the residuals (to see how the lags show correlation), and the values of the residuals to check if they look stationary. The function also returns the p-value of the Ljung-Box test which sees the statistical significance of the stationarity of the variable. We select the method that gives the most stationary uncorrelated residuals. We also perform a kpps.test (Kwiatkowski-Phillips-Schmidt-Shin) and adf.test (augmented Dickey–Fuller test) to

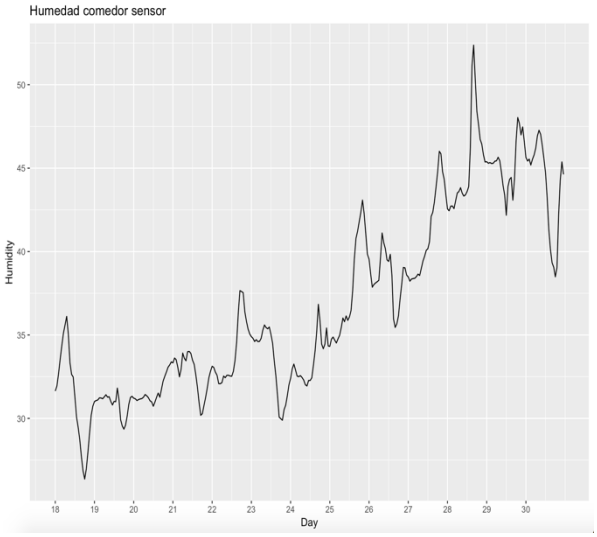
triple check the stationarity (mean, variance and auto-correlation constant through time) of the residuals. Moreover, once we select a method, if it is an arima method, we check its invertibility through the unit root test, seeking to see if the coefficients of the arima method are well fitted into the unit root circle. To perform this test, an autoplot of the arima method graphically summarizes the unit roots.

- The selected method is used with the forecast method in R, in which we specify the forecasting horizon in hours as an argument. We use h=1, h=24 and h=120 as they are our forecasts of interests. We extract the values as a plot and as a dataframe, more precisely as a tibble from the package "tibble".
- Finally, we use the method tsCV() to perform a times series cross validation. To be able to use this function we need to create another function that returns an object consisting of the forecast of the method chosen with the time horizon we are interested in. This method allows us to have a score which represents the difference between the forecasted values and the measured values, that we call "e" as "error", from a given variables with training sets of different lengths. This score allows us to compute the RMSE to have an evaluation of the model.

RESULTS

1) Humedad Comedor Sensor

Exploratory analysis



When applying the autoplot method to the series (which we call humcomsens here as an abbreviation of humedad comedor sensor), we can clearly spot an upward trend in this time series going from around 30 to more than 45. We can already say the series is non-stationary and the variance seems quite random. Moreover, after creating a seasonal plot (Figure 4: Seasonal plot of Humedad comedor sensor. Each color is a day and not a year.), we do not see a clear seasonal pattern, but we do see that each day from hour 1 to hour 7, the humidity seems to be very stable. Thus, there might be a seasonal pattern of 24 hours.

From midnight to 6AM, the humidity does not vary much as we can see with the ggsubseriesplot method (Figure 5: Seasonal subseries of Humedad Comedor Sensor). However, as the day unfolds, there seems to be a varying variance from past noon on. Some hours seem to be very alike and others vary much more as hour 17, 18 and 19 which display the most variance. This might indicate that we will need a transformation before applying a forecasting method to stabilize the variance.

In order to have a better idea about seasonality and the patterns displayed, we use time series decomposition: additive, multiplicative and stl decompositions. When comparing the three, stl appears to be the best (Figure 6: Multiplicative decomposition of Humedad_Comedor_Sensor, Figure 7: Additive decomposition of Humedad_Comedor_Sensor, Figure 8: STL decomposition of Humedad_Comedor_Sensor). Indeed, stl decomposition seems to leave less structure in the remainder.

Because of the changing variance, we decide to apply a BoxCox transformation before testing all the methods that are not arima, and the variance looks more stable (Figure 9: BoxCox transformed

Humedad_Comedor_Sensor). The BoxCox transformed series is called trhumcomsens here. When applying the STL decomposition to the transformed series (trhumcomsens), the remainder is smaller, but the structure left is the same. Depending on the function tested, we will use either humcomsens or trhumcomsens.

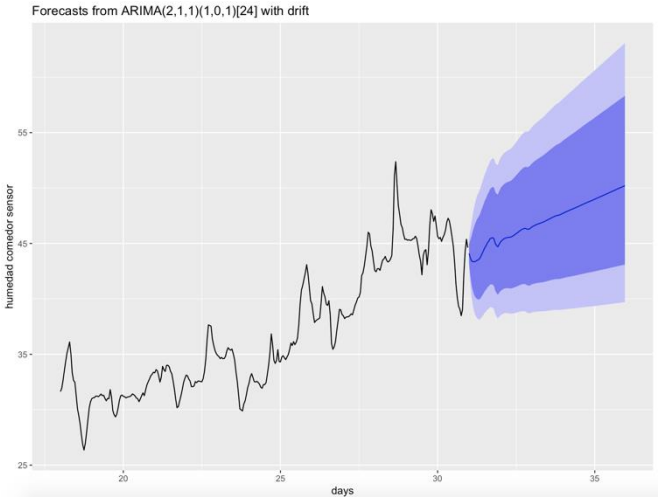
Forecasting methods

Because of the seasonality that might be revealed by this series, the contender methods we decide to test are ARIMA (autoregressive integrated moving average), ETS (error, trend, seasonality) and HW (Holt-Winter).
Using the trhumcomsens, the Box-Cox transformed series, we select it to test ets and hw, whereas the untransformed series is used with auto.arima as auto.arima performs the optimal transformation with our arguments.
For the holt-winter method we choose h=24 as seasonality because it is the most plausible one, when looking at the autoplot. The R command used is*: hw<-hw(trhumcomsens, h=24).*
For the arima method, we take an auto.arima with an automatic BoxCox transformation, a stepwise false comparison (to compare every possible models and take the best one), an automatic back-transformation of the result. The R command is*: humidity1arima<-auto.arima(humcomsens, stepwise = FALSE, lambda="auto", biasadj=TRUE, allowdrift=TRUE).* Finally for the ets model, we choose a damped option with the following command: *ets<-ets(trhumcomsens, damped = TRUE).*We compare the respective AIC of the methods: The AIC of the auto.arima method (which gives us an arima with (2,1,1)(1,0,1)[24] parameters, confirming our intuition of a daily seasonality) is -1410,426. The AIC of the hw method is -238,94 and the AIC from the ETS method is -480,4923. Looking at these results, the arima model seems to be the best fitted. To be sure of this, we compare their residuals:
The residuals for the hw method do not look normally distributed and there are too many correlated lags (outside of the limits) (Figure 10: Residuals of the HW method on the CoxBox transformed Humedad_Comedor_Sensor.). The residuals from the ets are more normally distributed and look stationary, but there are still 6 lags out of 72 lags outside of the limits. This indicate a correlation of 8,33% which is more than the ideal (under 5%) (Figure 11: Residuals of the ETS method on the BoxCox transformed Humedad_Comedor_Sensor).
Finally the residuals from the ARIMA(2,1,1)(1,0,1)[24] have only 2 lags outside the limits with a stationary look and a normally distributed residual (Figure 12: Residuals from ARIMA(2,1,1)(1,0,1)[24] of Humedad_Comedor_Sensor). The arima model seems again to be the best. To make sure the residuals are stationary and do not leave any trend nor seasonality, we compute the Pacf of the residual (ggPacf) (Figure 13: ggPacf of the residuals resulting from ARIMA(2,1,1)(1,0,1)[24] for Humedad_Comedor_Sensor.) and see that there are also only two lags outside of the limits, which confirms the model seems well suited for the forecast of this variable. As auto.arima does not test for invertibility, we need to see if our model is possible with the unit root test, we get that all are within the circle, as desired.(Figure 14: Unitroot test of the selected ARIMA(2,1,1)(1,0,1)[24] method for Humedad_Comedor_Sensor.)
Finally, to be sure of the stationarity of the residuals we also apply the adf and kpss tests with the following codes: adf.test(humidity1arima$residuals), kpss.test(humidity1arima$residuals). The adf test (augmented Dickey-Fuller) yields that the p-value is smaller than the printed p-value. As the null hypothesis of the test is non stationarity of the residual, we reject the null hypothesis and we conclude that residuals are stationary (p-value of 0,01). For the KPSS test, the null hypothesis is stationarity and we get a p-value of 0,1, which is greater than the printed p-value, meaning we do not reject the null hypothesis. Both tests are coherent with each other and the arima model is selected for the forecasting.

Forecasted values

Graphically, we cannot see the forecast for one hour as it is only a point. For this reason, we only represent the forecasts for 24 hours and 5 days. For 24 hours we see that the prediction interval is cone

shaped and supports the growing trend we see in the whole series (Figure 15: 24 hour forecast of Humedad_Comedor_Sensor using the ARIMA(2,1,1)(1,0,1)[24] with prediction intervals.).



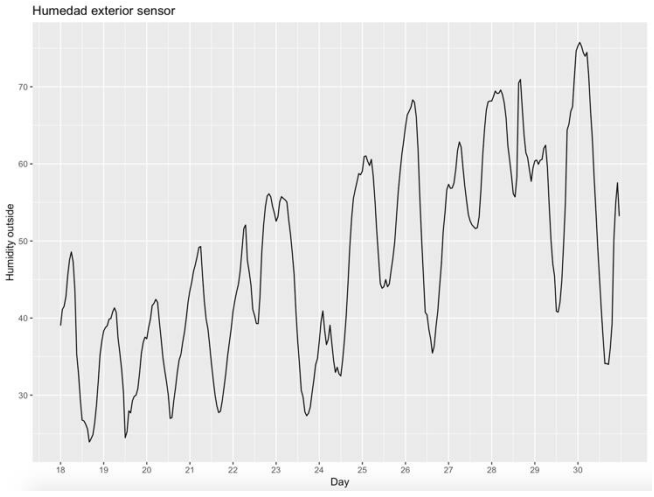
For 120 hours (5 days), we see the trend and the cone shaped prediction interval but the seasonality and patterns are completely dampened as the incertitude grows, the ARIMA converges towards the trend of the series. We can also look at the tibble table for the 1-hour and 24-hour forecasts (Figure 16: 1 hour forecast of Humedad_Comedor_Sensor with the selected ARIMA(2,1,1)(1,0,1)[24] and Figure 17: 24 hours forecast of Humedad_Comedor_Sensor with the selected ARIMA(2,1,1)(1,0,1)[24]). It is as we would expect, the forecast for 1h is the same as for the 1st hour of the forecast for 24h with the same model .The numerical table of the forecast for the next 5 days is too long to be included here, but the plot representation above gives us a good idea of the behavior of this forecast.

Evaluation with tsCV

We first create a function which turns the forecast from the auto.arima method into an object. Then, we use it with a selected forecast horizon. This allows us to use tsCV() to extract the error *e* depending on the series, the horizon and the object precedingly created as follows:

far2 <- function(x, h){forecast(auto.arima(x, step = F, lambda="auto", biasadj=T, allowdrift=T), h=h)}
e <- tsCV(humcomsens, far2, h=1)
We use the error to get the RMSEs of each forecast .
For the 1 hour forecast we get:
RMSE = sqrt(mean(e^2, na.rm=TRUE)) = 0.6959974

For the 24 hours forecast we get:
g<-tsCV(humcomsens, far2, h=24)
RMSE =sqrt(mean(g^2, na.rm=TRUE)) = 3.99
For the 120 hours forecast we get:
f <- tsCV(humcomsens, far2, h=120)
RMSE= sqrt(mean(f^2, na.rm=TRUE))= 13.4834



2) Humedad Exterior Sensor

Exploratory analysis

When applying the autoplot method to the series which we call humextsens here as an abbreviation of humedad exterior sensor, we can clearly spot an upward trend in this time series going from around 24% to more than 70%. We can already say the series is non-stationary and the variance seems to increase. It also seems like there is a 24-hour seasonality.

We do see a clear daily seasonal pattern with ggseasonplot but day 29 seems to go to the opposite direction, the humidity seems to always decrease from hour 0 to hour 13 and then goes up again. (Figure 18: Seasonal plot of Humedad_Exterior_Sensor.). In the ggsubseries plot(Figure 19: Subseasonal plot of Humedad_Exterior_Sensor), we see that the variance is not stable through the hours. Hours 17, 18 and 19 display the most variance. We also notice there is a clear seasonality with a slight increase from hour 1 to 6, a big decrease from hour 7 to 13 and then an increase. This last increase goes along with an increase in variance. This indicates that we need a transformation to stabilize the variance. In order to have a better idea about seasonality and the patterns displayed, we use seasonal decompositions : additive, multiplicative and stl decompositions. When comparing the three, it seems that stl is the best as for the variable humedad comedor sensor. Indeed, stl decomposition (Figure 20: STL decomposition of Humedad_Exterior_Sensor) seems to retain the least structure in the remainder. Because of the changing variance, we apply a BoxCox transformation, and the variance looks more stable (Figure 21: BoxCox transformed Humedad_Exterior_Sensor). The BoxCox transformed series is called trhumextsens here. When applying the STL decomposition to the transformed series (trhumextsens), the remainder is smaller, but the structure displayed is the same . Depending on the function tested, we will use either humextsens or trhumextsens.

Forecasting methods

Because of the seasonality in this series, the contender methods we use to test are arima, ets and hw. Using the trhumextsens series, we use ets and hw, whereas the untransformed series is used with auto.arima as auto.arima performs the optimal transformation with our arguments.
For the holt-winter method we choose h=24 as a seasonality because it is the one that we notice in the autoplot. The R command used is: *hw<-hw(trhumextsens, h=24).* For the arima method, we take an auto.arima with an automatic BoxCox transformation, a stepwise false comparison (to compare all possible models and take the best one) , an automatic back-transformation of the result.
The R command is: *humidity2arima<-auto.arima(humextsens, stepwise = FALSE, lambda="auto", biasadj=TRUE, allowdrift=TRUE)*. Finally for the ets model, we choose a damped option with the following command: *ets<-ets(trhumextsens, damped = TRUE).* We compare the respective AIC of the methods:

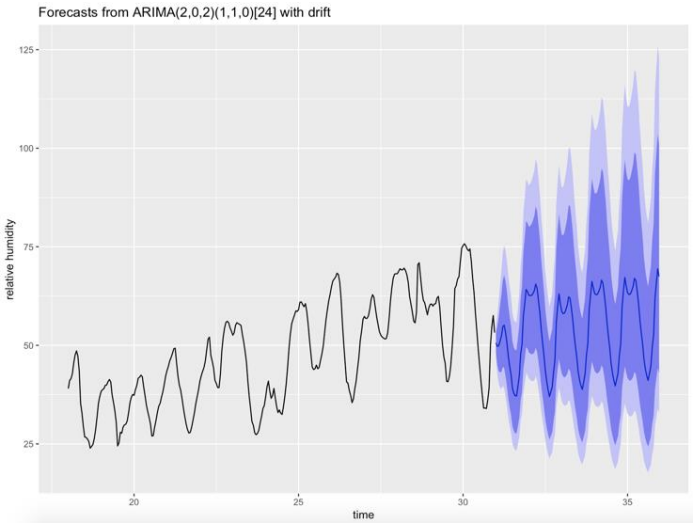
- AIC of ARIMA(1,0,2)(2,1,0)[24] is -396,3
- AIC of the HW method is 515,3211.
- AIC from the ETS method is 491,30.

Looking at these results, the arima model seems to be the best fitted. To be sure of this, we compare their residuals. The residuals for the hw method have too many correlated lags (outside of the limits) (Figure 22: Residuals from the Holt-Winters' method on the BoxCox transformed Humedad_Exterior_Sensor.). The residuals from the ets are more normally distributed and look stationary, but there are still too many lags outside of the limits (Figure 23: Residuals from ETS (M,Ad,N) on the BoxCox transformed Humedad_Exterior_Sensor). Finally the residuals from the arima(1,0,2)(2,1,0)[24] have 5 lags outside the limits with a stationary look and a normally distributed residual. The arima model seems again to be the best, but 5 lags out of 72 gives us an autocorrelation that is still too high. The p-value of the Ljung Box test of 0.09 is high also. We need to try other models. We try an auto.arima with no drift allowed, nor biasadjust with the following code: *model1<-auto.arima(humextsens, lambda="auto").* We get an arima (2,0,2)(1,1,0)[24], with an AIC of -369,759. The residuals look the best with 3 lags out of the limit, indicating a low autocorrelation (Figure 24: Residuals from ARIMA(2,0,2)(1,1,0)[24] of Humedad_Exterior_Sensor). They also look normally distributed and stationary. To make sure the residuals are stationary and do not support any trend nor seasonality, we compute the Pacf of the residual (ggPacf) and see that there are also only two lags outside of the limits (Figure 25: ggPACF of the ARIMA(2,0,2)(1,1,0)[24] model for Humedad_Exterior_Sensor) , which confirms the model is well

suited for the forecast of this variable. Once again, we notice that all unit roots are within the unit circle (Figure 26: Unitroot test of ARIMA(2,0,2)(1,1,0)[24] model for Humedad_Exterior_Sensor).

Finally, to make sure of the stationarity of the residuals we also apply the adf and kpss tests with the following codes: adf.test(model1$residuals), kpss.test(model1$residuals). We get for the adf (augmented Dickey-Fuller) test that the p-value is smaller than the printed p-value. As the null hypothesis of the test is non stationarity of the residual, we reject the null hypothesis and our residuals are thus stationary. For the KPSS test the null hypothesis is stationarity, we get a printed p-value of 0,1. meaning we do not reject the null hypothesis. Both tests are coherent and the arima model is selected for the forecasting.

Forecasted values



Graphically, the 1-hour forecast is not visible. For this reason, we only represent the 24-hour and 5-day forecast. For 24 hours we see that the prediction interval is cone shaped and follows the growing trend we see in the whole series (Figure 27: 24 hours forecast of Humedad_Exterior_Sensor using ARIMA(2,0,2)(1,1,0)[24] with the 80% and 95% prediction intervals computed in blue and light blue respectively.).

For 5 days in advance , we see the trend and the cone shaped prediction interval with a clear daily seasonality patterns. We can interpret that the seasonality patterns are very regular, which is not so clear looking at the series before the forecast. The remainders and errors will probably give more irregularity to the real measurements curves. We see the tibble for the 1 hour forecast and for the 24 hours forecasts (Figure 28: 1-hour forecast of Humedad_Exterior_Sensor using ARIMA(2,0,2)(1,1,0)[24] and Figure 29: 24 hours forecast of Humedad_Exterior_Sensor using ARIMA(2,0,2)(1,1,0)[24]).As we expect, the 1-hour forecast is the same as the 1st hour of the 24-hour forecast. . The numerical forecast table for the 5 day-forecast is too long to be included here, but you can see its behavior above in the graphical representation.

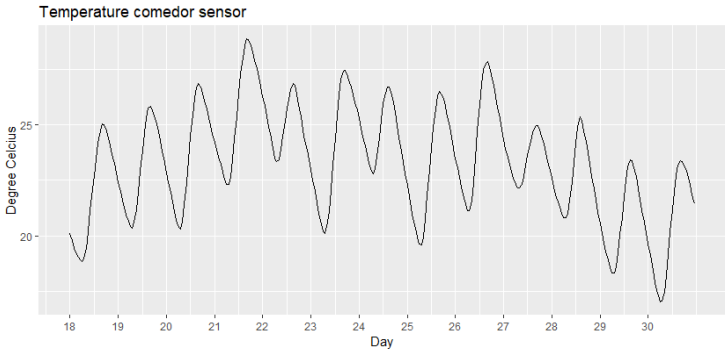
Evaluation with tsCV

Using the same function as for the first variable:

far3 <- function(x, h){forecast(auto.arima(x, lambda="auto")),h=h}
e1<- tsCV(humextsens, far3, h=1)
We use the error to get the RMSEs of each forecast .
For the 1 hour forecast we get:
RMSE = sqrt(mean(e1^2, na.rm=TRUE)) =2.242156
f1 <- tsCV(humextsens, far3, h=5)
For the 24 hours forecast we get:
g1<-tsCV(humextsens, far3, h=24)
RMSE =sqrt(mean(g1^2, na.rm=TRUE)) = 5.637178
sqrt(mean(g1^2, na.rm=TRUE))
For the 120 hours forecast we get:
f 1<- tsCV(humextsens, far3, h=120)
RMSE= sqrt(mean(f1^2, na.rm=TRUE))= 11.486

3) Temperature Comedor Sensor

Exploratory analysis



In order to have a first overview of the variable temperature comedor sensor (tempcomsens) we use the command autoplot(). As we can notice, it seems to be a strong seasonality in temperature throughout the series. The series seems to have a slight increasing, then decreasing trend. It does not seem stationary but does have a stable variance.

After using the seasonplot() (Figure 30 : Seasonal plot of Temperature_Comedor_Sensor) and the ggsubseriesplot() (Figure 31 : Subseasonal plot of Temperature_Comedor_Sensor) commands, our intuition of seasonality is confirmed. There is a clear sinusoidal curve throughout the day. The peak for each day is around 4pm and the dip is around 7am.

After running the ggAcf command (Figure 32 : ACF of Tempereature_Comedor_Sensor), we remark that the first few lags are heavily correlated but seasonality is also important as the 24th lag (previous and subsequent lags too) is heavily and positively correlated. This is also the case for the 48th lag. We deduce that there is a clear daily seasonality. To get more insights into this variable, we decide to decompose the time series into a trend component, a seasonal component and remainder. We decide to use several methods namely additive decomposition, multiplicative decomposition and a stl decomposition (Figure 33 : STL decomposition of Temperature_Comedor_Sensor).The STL method seems to yield the best results. First, the remainder are less important in absolute terms. Secondly, unlike the traditional decomposition method, it can be robust to outliers (which we specified in our analysis). Finally, unlike SEAS and X11 methods, which we did not use here, STL can handle hourly seasonality which is very practical in our analysis. We still notice there is seasonality left in our remainder. This is something we will have to handle before starting the modelling process.

Forecasting methods

We decide to test arima and ets models for the variable temperature comedor sensor as we have a clear seasonal pattern. We use a seasonal differentiation in order to handle the seasonality left in the remainder. We fit a model on tempcomsens (temperature comedor sensor not transformed) using the auto.arima() function; *ARIMA_auto_tempcomsens <- auto.arima(tempcomsens, stepwise = FALSE, biasadj = TRUE, allowdrift = TRUE, lambda = "auto")* and we get an ARIMA(2,1,0)(2,1,0)24.

We fit another model using the ets() function on the seasonally adjusted data;

ets_temp <- ets(seasonaldifftemp, model = "ZZZ", damped = NULL, alpha = NULL, beta = NULL, gamma = NULL, phi = NULL,lambda = "auto", biasadj = TRUE, ic = c("aicc", "aic", "bic") and get an ETS(A,Ad,N). We compute the AIC of each model to compare them and decide which one should be kept. The AIC of the arima model is equal to -790.0292 while the one of the ets model is 557.2958. According to this criterion, the arima model should be chosen. We plot their residuals to be sure of this conclusion (Figure 34 : Residuals for ARIMA(2,1,0)(2,1,0)[24] for Temperature_Comedor_Sensor, Figure 35 : Residuals for ETS(A, Ad, N) for Temperature_Comedor_Sensor). The residuals of the ARIMA(2,1,0)(2,1,0)24 seem stationary and random. However, we still notice a slight pattern that can be observed between the days 20 to 23. The ACF graph shows that there is no more strong autocorrelation as only one lag goes out of the interval. We can see

that the residuals are following a normal distribution. When we analyze the residuals of the ets model, we can observe that they still show some autocorrelation as there are 6 lags out of the interval. Their distribution looks stationary and they seem to be normally distributed. These results show that the arima model is slightly better. We check for invertibility as the function auto.arima does not test for it (

Figure 36 : Unitroot test of ARIMA(2,1,0)(2,1,0)[24] model for Temperature_Comedor_Sensor) . We can confirm that all the points are within the circle.

Finally, in order to check the stationarity of the residuals, we perform an adf.test and a kpss.test. We get a p-value equal to 0.01 for the adf test. We therefore reject the null hypothesis which means that the residuals are stationary. For the kpss test, the p-value is 0.1. The null hypothesis is not rejected which leads to the same conclusions as the previous test. We can therefore deduce that our model is reliable and can be used for forecasting values.

Forecasted values

We can see the forecasted values in the appendix (Figure 37 : Short range forecast of Temperature_Comedor_Sensor (h=1), Figure 38 : 24 hours forecast of Temperature_Comedor_Sensor) and the graph for the 24 hours forecast (

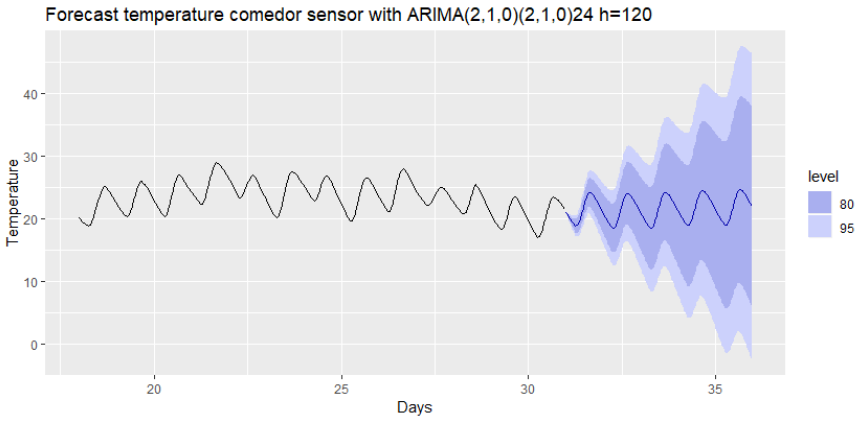


Figure 39 :Forecast of Temperature_Comedor_Sensor for 24 hours for ARIMA(2,1,0)(2,1,0)[24] method with the 80% and 95% prediction intervals).

Here is the graphical representation of the forecast obtained for 5 days.

The strong seasonality is still very present and we can possibly notice a very slight increasing trend. Most notably, the confidence interval is very cone shaped, showing a growing incertitude of the forecast.

Evaluation with tsCV

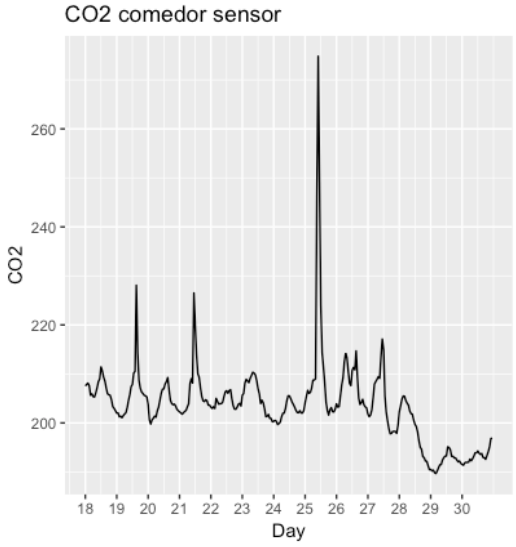
Using the same function as before, we run the following code. As we predict for longer ranges, the confidence intervals and the RMSE increase.

```
CvARIMA <- function(x,h){forecast(auto.arima(x, stepwise = FALSE), h=h)}
ARIMAcv1 <- tsCV(tempcomsens, CvARIMA, h=1)
ARIMAcv24 <- tsCV(tempcomsens, CvARIMA, h=24)
ARIMAcv120 <- tsCV(tempcomsens, CvARIMA, h=120)
```

```
RMSE1 <- sqrt(mean(ARIMAcv1^2, na.rm=TRUE))
RMSE24 <- sqrt(mean(ARIMAcv24^2, na.rm=TRUE))
RMSE120 <- sqrt(mean(ARIMAcv120^2, na.rm=TRUE))
RMSE1 = 0.1300184
RMSE24 = 2.848945
RMSE120 = 13.68994
```

4) CO2 Comedor Sensor

Exploratory analysis



We can first see that CO2 time series is not stationary as the variance changes a lot through time and we can spot a little decreasing trend from the day 26 on. There is a very high peak on April, the 25th. We can also spot a seasonality per day, where there is a peak from 9am to 2pm (Figure 40 : seasonal plot of C02_Comedor_Sensor, Figure 41 :Subseasonal plot of CO2_Comedor_Sensor). However this seasonality pattern seems weak.

In the ACF plot (Figure 42 : ACF of CO2_Comedor_Sensor), we can find a strong positive correlation in the first few lags. Moreover, there is only positive correlations. Thus, there is no stationary state.

Using the same methods for the decomposition, we used the STL decomposition (Figure 43 : STL decomposition of C02_Comedor_Sensor).

As the variance is changing, we applied a BoxCox transformation, the variance looks the same. We called it trco2comsens. With the STL decomposition, troco2comsens looks the same as CO2comsens.

Forecasting methods

We decide to test arima and ets models for the variable co2 comedor sensor.

We get an ARIMA(1,1,2)(2,0,0)24 with an AIC of -5068.588 and ETS(A,N,N) with an AIC of -4120.533. Based on this result, we select the ARIMA model. To be sure of this choice, we make an analysis of the residuals (Figure 44 : ARIMA(1,1,2)(2,0,0)[24] residuals of CO2_Comedor_Sensor, Figure 45 : ETS(A,N,N) residuals for CO2_Comedor_Sensor).

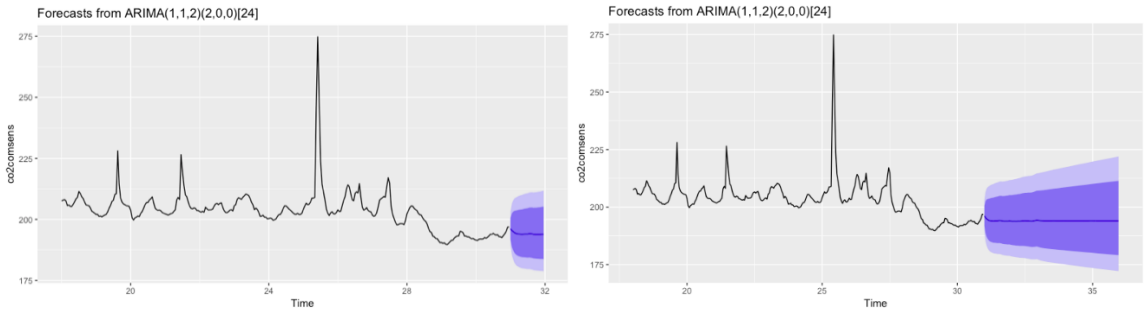
The residuals from ARIMA seems stationary, random and following a normal distribution. Only two lags go out of the ACF graph interval. Residuals from ETS display more autocorrelation with 6 lags out of the ACF graph interval. The distribution of the ets model residuals and ARIMA's residuals are following the same characteristics: stationary, random and normally distributed. We can conclude that ARIMA is better than ETS. To check for invertibility, we used the autoplot function (Figure 46 : Unitroot test of CO2_Comedor_Sensor). We can confirm that all the points are within the unit circle.

With the ADF test, we can spot that the p-value is smaller than the printed p-value. We then reject the null hypothesis, and thus the residuals are stationary. The KPSS test shows us that we do not reject the null hypothesis of the stationarity over the residuals. KPSS and ADF agreed on the stationary state over the residuals.

Forecasted values

Forecasts for 24 hours and 120 hours of CO2 comedor sensor are shown below. In figure 47 (

Figure 47 : 24 hour forecast of CO2_Comedor_Sensor using ARIMA(1,1,2)(2,0,0)[24]) we can see the numerical values of the 24h forecast.



We can see that the forecast is almost linear. It might be because there is no seasonality and most the variation is not explained by it, so ARIMA converges to the trend to minimize the sum of square error of the forecast.

Evaluation with tsCV

Using the same function as before, we run the following code:

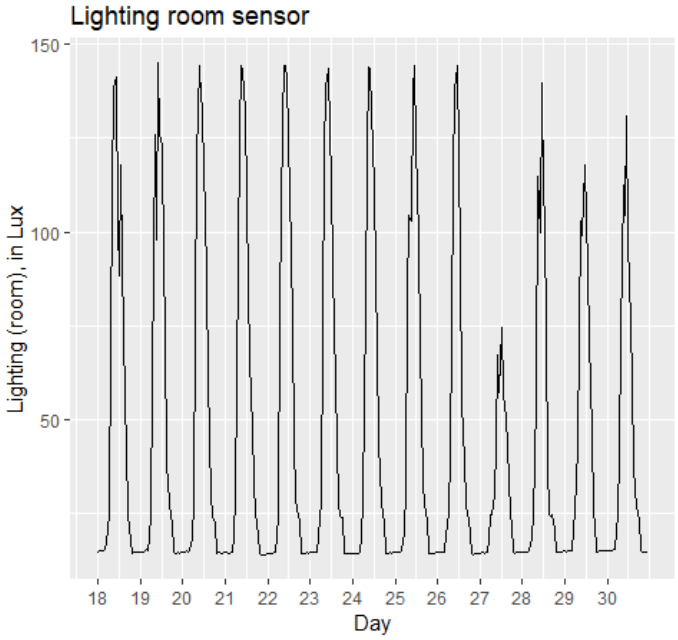
```
CvARIMA <-
function(x,h){forecast(auto.arima(co2comsens,
stepwise = FALSE, biasadj = TRUE, lambda
= "auto"), h=h)}

ARIMA_co2_cv120 <- tsCV(co2comsens,
CvARIMA, h=120)
ARIMA_co2_cv1<- tsCV(co2comsens,
CvARIMA, h=1)
ARIMA_co2_cv24 <- tsCV(co2comsens,
CvARIMA, h=24)
```

```
RMSE1 <-  sqrt(mean(ARIMA_co2_cv0^2,
na.rm=TRUE))
RMSE24 <-  sqrt(mean(ARIMA_co2_cv24^2,
na.rm=TRUE))
RMSE120 <-  sqrt(mean(ARIMA_co2_cv1^2,
na.rm=TRUE))
RMSE1: 11.03737
RMSE24: 12.4053
RMSE120 : 12.54593
RMSE is high, it looks hard to forecast this
parameter.
```

The values found here for the RMSE are higher than the preceding variables and indicate a strong error that begins for the first forecast even for 1 hour. More is discussed in the discussion and conclusion

5) Lighting Habitacion Sensor

Exploratory analysis



Regarding lighting we see a clear variation throughout the day but also strong seasonality (Figure 48 : seasonal plot of Lighting_Habitacion_Sensor, Figure 50 : Subseasonal plot of Lighting_Habitacion_Sensor). The amplitude varies from just below 15 during the night to 145 during the day. When we run a seasonal plot, all the observations for the several days evolve generally in the same way although there are some days with less light. The lighting intensity starts to increase around 5AM, peaks from 10 AM to 12 PM and comes back to its initial level at 7PM. Day 28 seems to be an exception, however. The mean lighting for each hour also confirms the existence of seasonality.

The ACF (Figure 51 : ACF of Lighting_Habitacion_Sensor) plot confirms daily seasonality, with very correlated lags (Figure 49 : Lagplots of Lighting_Habitacion_Sensor). The 24th and 48th lags are also very significant which demonstrates the recurring pattern. We would go for a STL decomposition (Figure 52 : STL decompositon of Lighting_Habitacion_Sensor) as the remainder seems to be less structured than in the additive and multiplicative decompositions.

Forecasting methods

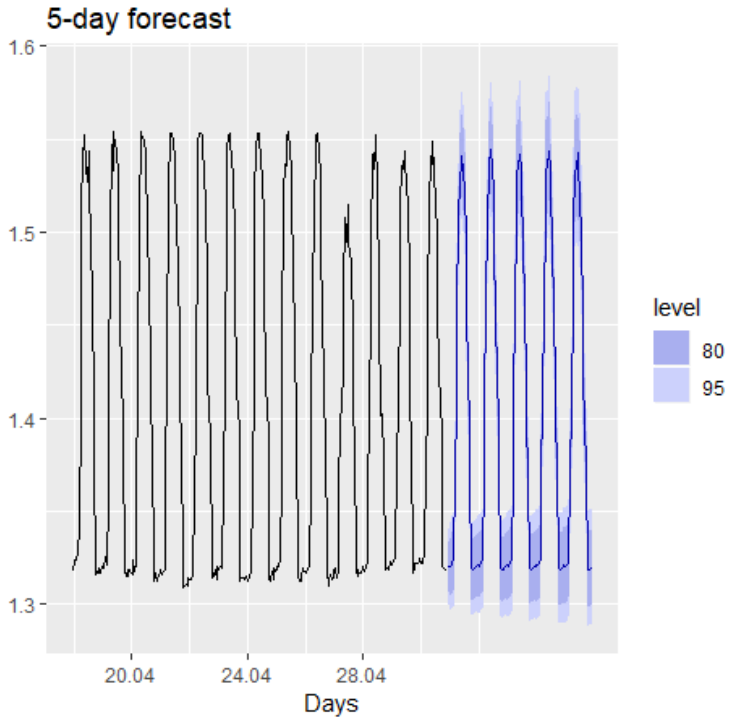
Again, we decide to test to class of models namely ARIMA and ETS. After applying a Box-Cox transformation (Figure 53 : BoxCox transformed Lighting_Habitacion_Sensor) , we use the auto.arima() and ets() methods. We get the following results:

- ARIMA(1,0,0)(1,1,2)[24] AIC = -1698.82
- ETS(A,Ad,A) AIC = -878.57

We directly notice that forecasts with ETS(A,Ad,A) are of poor quality because the residuals are much higher than with an ARIMA(1,0,0)(1,1,2)[24]. If we compare the AIC of both models we get -1698.82 for the ARIMA and -878.57 for the ETS. We will therefore focus on ARIMA. When we check the residuals of the aforementioned ARIMA model we get the following results (Figure 54 : Residuals of ARIMA(1,0,0)(1,1,2)[24] method for Lighting_Habitacion_Sensor) . The residuals seem to look like white noise, although not perfectly as there are 2 big dips at the end of the series. When we analyze the ACF plot, we notice 3 bars out of 72 which go beyond the interval (dashed blue line). This would equal approximately 4.2% of the series, it is therefore below a 5% level. Looking at the density, we notice a very high concentration around the 0 mean. When we run the KPSS test to test stationarity, we get a 0.1 p-value with a warning (where it is mentioned that the p-value is actually greater than the one printed). Thus, we cannot reject the null hypothesis stating that our time series is stationary. We can therefore assume stationarity. The

unit root test (Figure 55: Unit root test for ARIMA(1,0,0)(1,1,2)[24] model for Lighting_Habitacion_Sensor) tells us that all the roots are invertible as they lie within the circle. Thus, the process is stable and will not explode.

Forecasted values:



After the analysis of our model, we move to the forecast itself. When we use the forecast() method we get the following graphs 5-day intervals respectively. The graphs for the 1-hour and 24-hour intervals are in appendix. We notice that the forecast look very much the same for every day. The increase and decrease is very well predicted. What is harder to predict is the top and the bottom, this is shown in a larger prediction interval. As we have a very repetitive series this makes it more easily predictable. (Figure 56: 1 hour forecast of Lighting_Habitacion_Sensor using ARIMA(1,0,0)(1,1,2)[24], Figure 57: 24 hours forecast of Lighting_Habitacion_Sensor with ARIMA(1,0,0)(1,1,2)[24] with the 80% and 95%, Figure 58: RMSEs score per variable with the selected methods) .

Cross validation with tsCV

Finally, we run a cross-validation for the 3 forecasts using the following code chunk.

```
far2 <- function(x, h){forecast(auto.arima(x, step = F), h=h)}
#1 hour in advance
e_1h <- tsCV(lightbc, far2, h=1)
#RMSE
rmse1 <- sqrt(mean(e_1h^2, na.rm=TRUE))
#24 hours in advance
e_24h <- tsCV(lightbc, far2, h=24)
#RMSE
rmse2 <- sqrt(mean(e_24h^2, na.rm=TRUE))
#5 days in advance
e_5d <- tsCV(lightbc, far2, h=5*24)
#RMSE
rmse3 <- sqrt(mean(e_5d^2, na.rm=TRUE))
rmse1 = 0.01958917
rmse2 = 0.1249433
rmse3 = 0.5791705
```

As the forecast interval widens, the RMSE becomes higher. This is intuitive because predicting further in the future is harder. However, compared with the other RMSE, this prediction gives the least error.

DISCUSSION AND CONCLUSION

For all variables, we select an ARIMA method to perform the best forecast. The use of auto.arima(), looking directly for the best possible model, is really helpful to get the lowest AIC possible. We find it consistently better than ETS and Holt-Winter for our specific variables. Of course, this is specific to our variables and other methods would be more effective on other measures.

The two variables with the least seasonality, namely: CO2 comedor sensor and humedad comedor sensor, display a forecast with a seasonality that decreases quickly to get linear. For CO2 comedor sensor, the forecast is completely horizontal. The interpretation behind this is that, because a lot of the variation of the CO2 measure is random, the ARIMA method forecast goes quickly to forecasting the trend as it is an ARIMA with one differentiation degree. If it had no differentiation degree, the forecast would quickly converge to the mean instead. This is a mathematical way to minimize the possible sum of square errors as the true measures will probably deviate from the trend randomly. The sum would be smaller with values on the trend than if a value is randomly much bigger or smaller than the true value. For humedad comedor sensor, we have the same process that is seen, except that the trend is increasing and not horizontal. The RMSEs (Figure 58: RMSEs score per variable with the selected methods) of both these variables computed through a cross-validation method start higher than the other variables even for the 1-hour forecast, and increase further for the longer forecasts. The uncertainty due to the random and non-seasonal changes is manifested very fast. It would mean that these measures would be harder to predict precisely even for 1 hour later using our methods, compared to predicting the other variables.

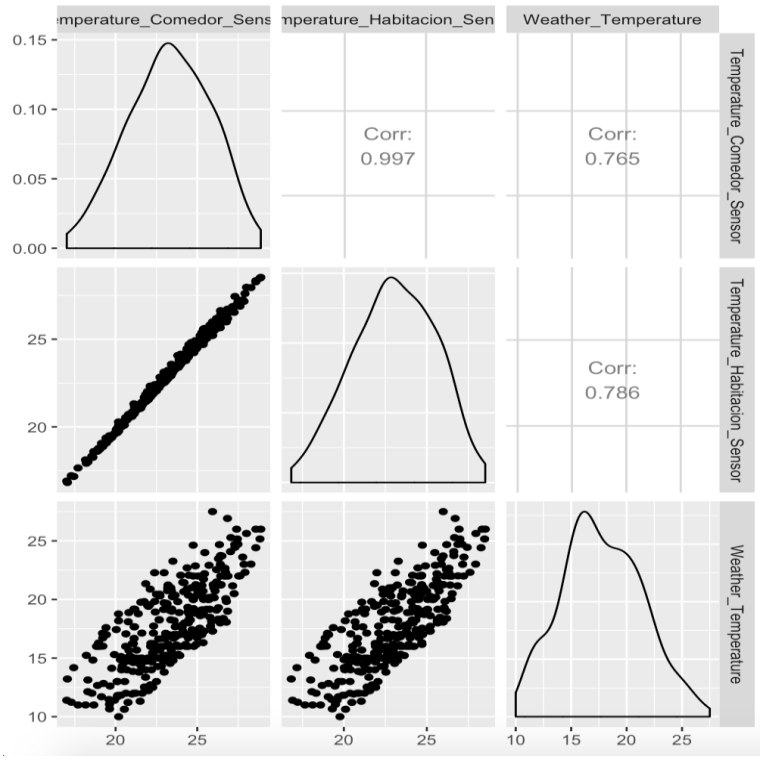
For a less "mathematical" interpretation, we thought that the values in the comedor, being a closed room, would depend on whether people are inside or not (for example, the weekend if more people are there, this would make sense that there is more CO2 and humidity produced by the breathing of the inhabitants). It would depend on how much people open the windows, which could depend on rain for example. This could be an interesting behavior to observe in a further exercise or research (for example use a dynamic regression model).

For the other variables, the most seasonal ones, lighting room sensor and temperature comedor sensor give us the best predictions with the smallest RMSEs tested by cross-validation for the 1-hour and the 24-hour prediction (Figure 58: RMSEs score per variable with the selected methods.). For the 120-hour prediction however, the RMSE of temperature comedor sensor dramatically increases. It is consistent with its prediction intervals which get very quickly larger in a large cone-shape, opposed to lighting habitacion sensor for which the prediction interval, still cone-shaped, gets larger at a slower rate. It is really striking that "lighting room sensor", in particular, has a very small error in the prediction compared with the others we get. Indeed, these variables have really small variation due to unpredictable changes, and both of them display small residuals in the STL decompositions (Figure 33 : STL decomposition of Temperature_Comedor_Sensor, Figure 53 : BoxCox transformed Lighting_Habitacion_Sensor), so the ARIMA method can minimize the error by repeating the seasonal pattern through time. Lastly, the variable humedad exterior sensor displays a pattern in between these two extremes of very weak seasonality and very strong seasonality.

Finally, some limitations of this study are due to a relatively small database first. Indeed, we can imagine meteorological data displaying different types of seasonality, probably monthly, quarterly, yearly etc. Moreover, 350 measures might not give us enough insight into what happens to this experimental house. Another limit comes from our limited techniques, and surely there must be other processes more efficient to provide predictions of better quality. To conclude, our study gives us a result that can hardly be used outside of the purpose of testing forecasting results, as there would be no way to use this forecast for a different point in time, a different geographical area, etc.

APPENDIX

Figure 1: Correlation of temperature measurements



Correlations between the variables Temperature_Comedor_Sensor (upper left), Temperature_Habitacion_Sensor (middle) and Weather_Temperature (upper right).

Figure 2: Correlations of CO2 measures and humidities

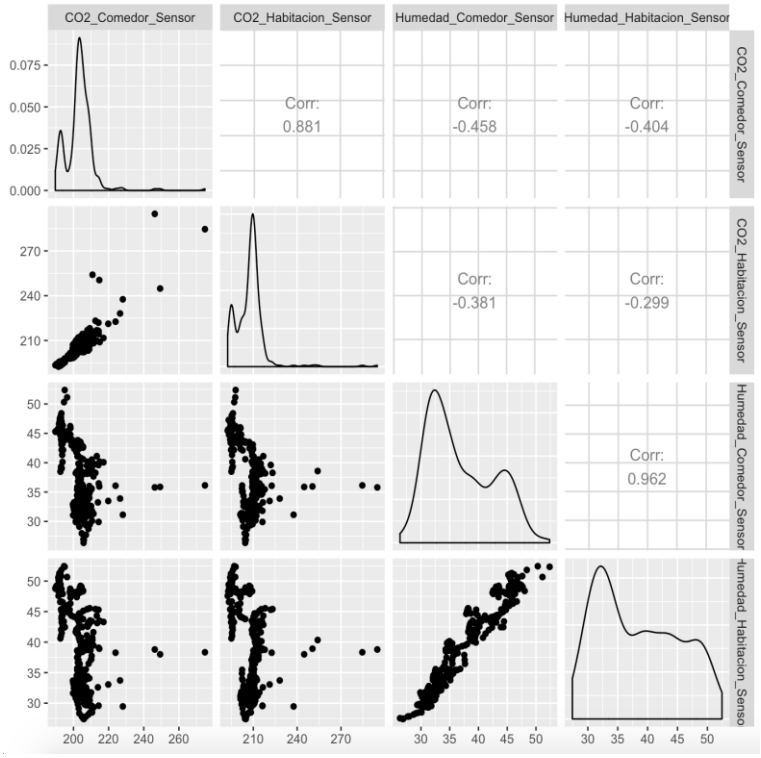


Figure 3: Correlation between our variables of interest

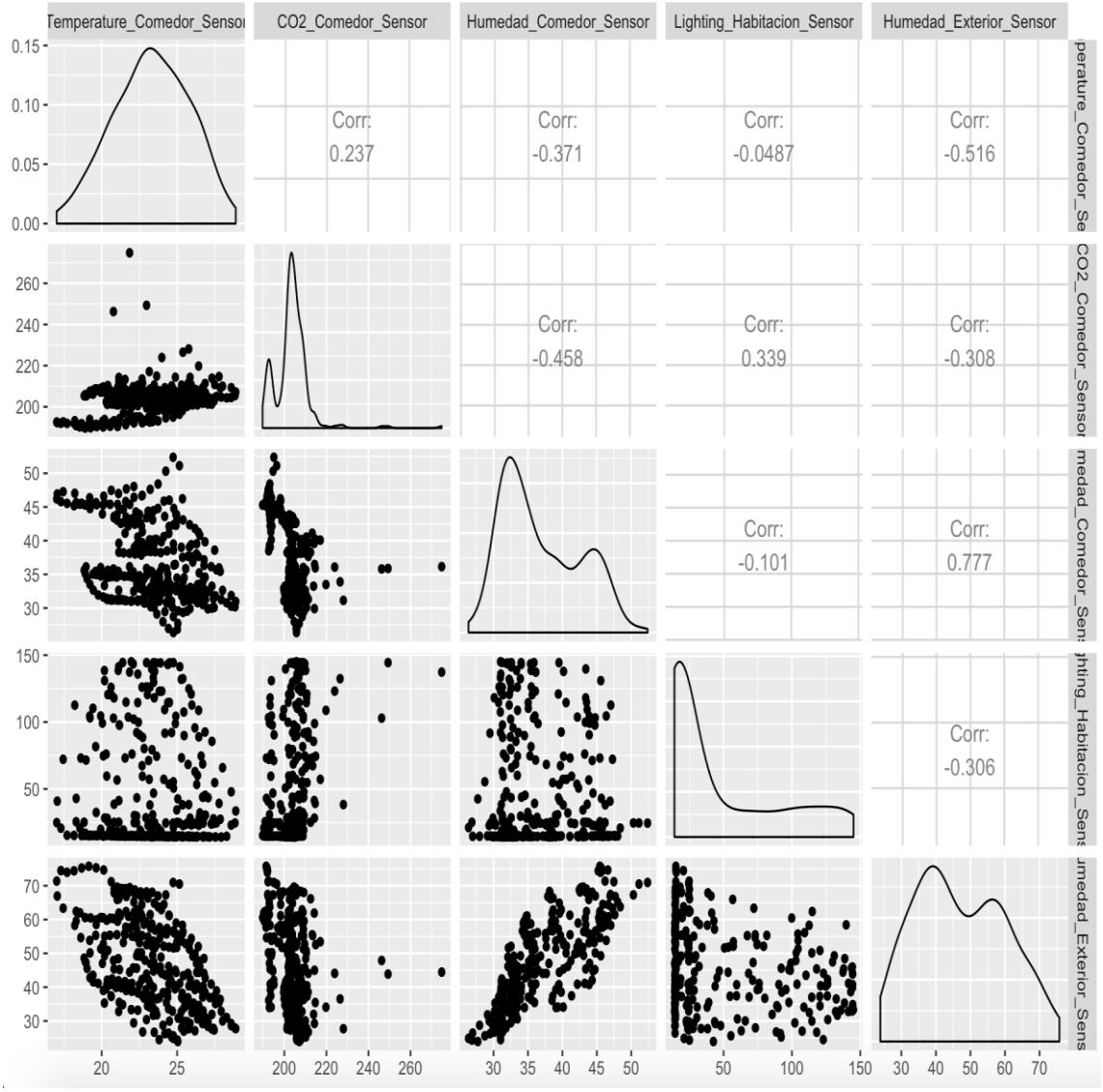


Figure 4: Seasonal plot of Humedad comedor sensor. Each color is a day and not a year.

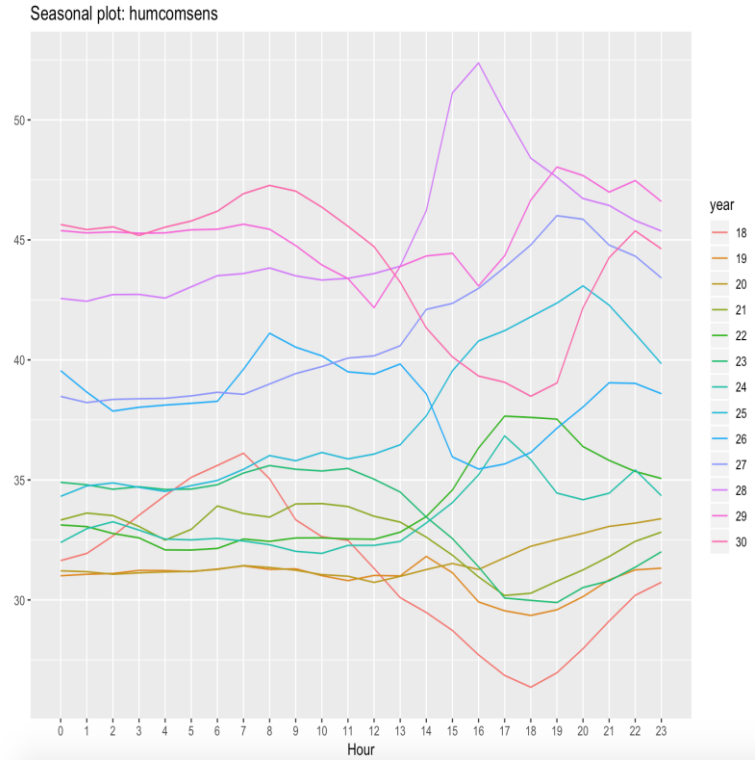


Figure 5: Seasonal subseries of Humedad Comedor Sensor

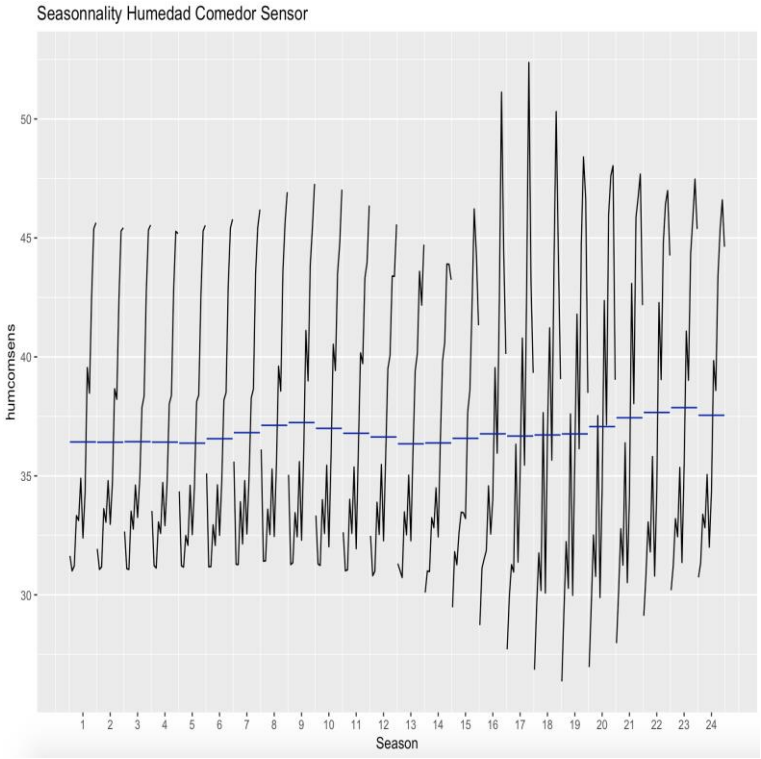


Figure 6: Multiplicative decomposition of Humedad_Comedor_Sensor

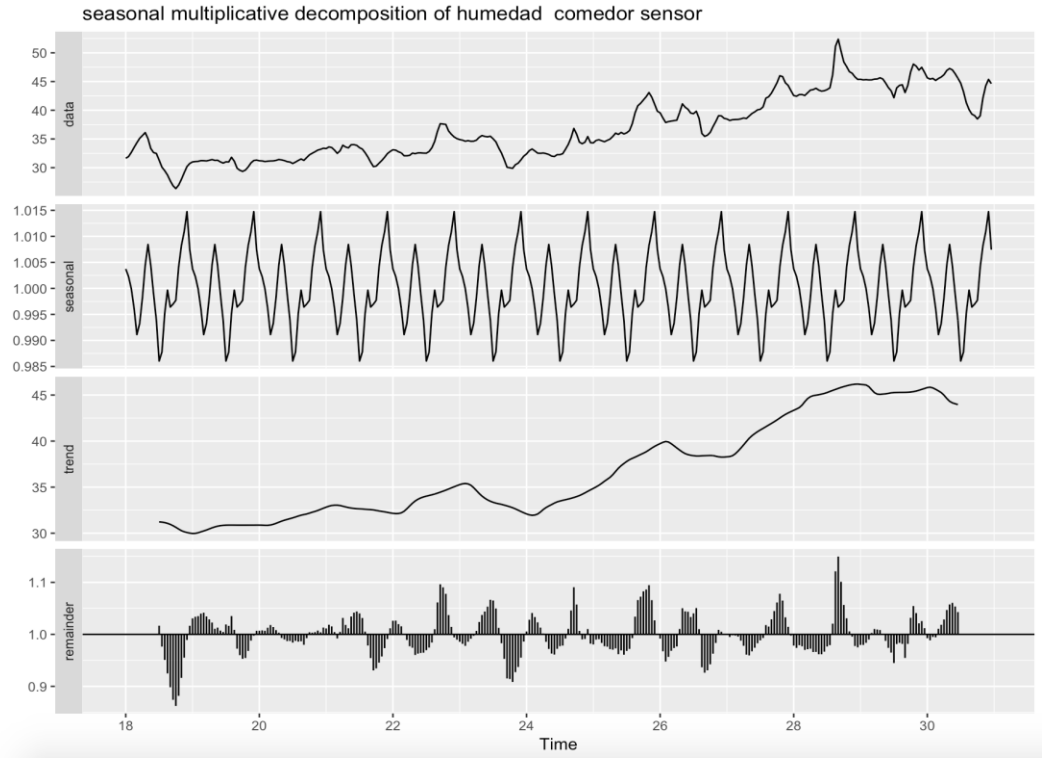


Figure 7: Additive decomposition of Humedad_Comedor_Sensor

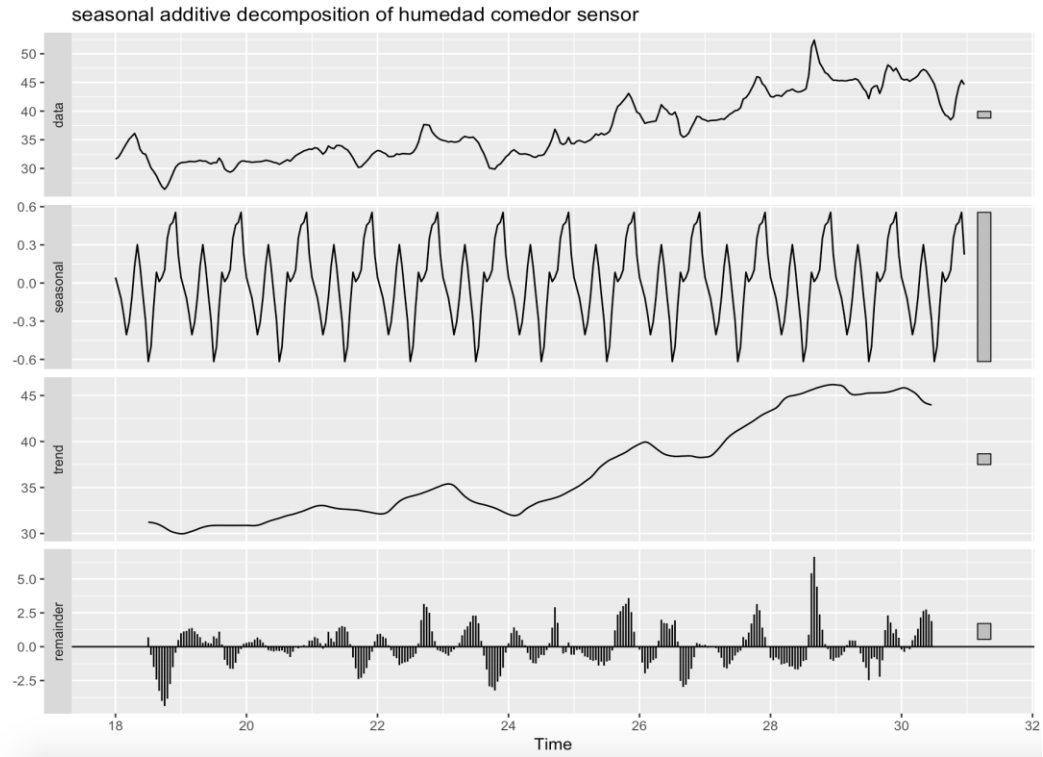


Figure 8: STL decomposition of Humedad_Comedor_Sensor

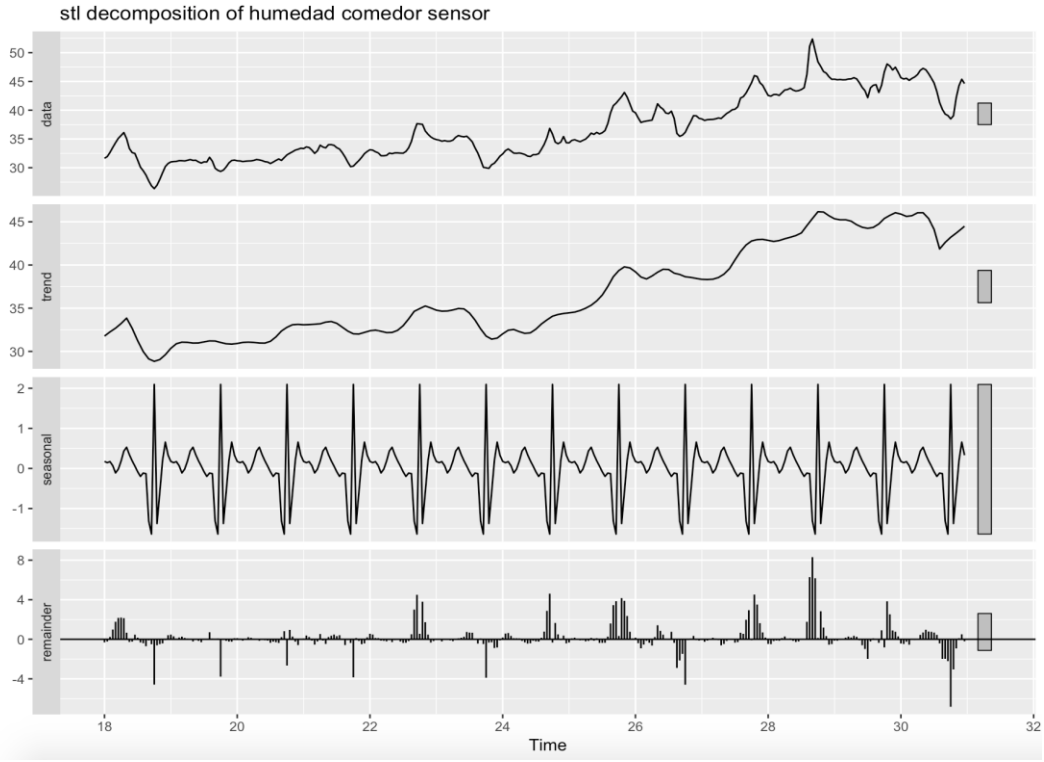


Figure 9: BoxCox transformed Humedad_Comedor_Sensor

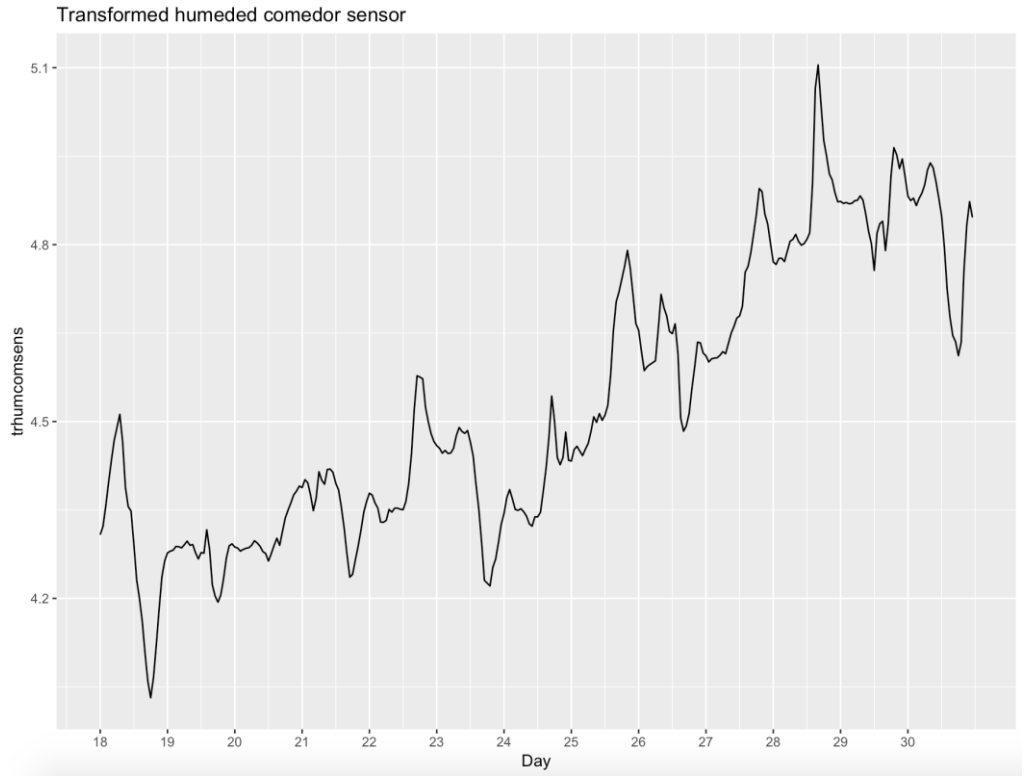


Figure 10: Residuals of the HW method on the CoxBox transformed Humedad_Comedor_Sensor.

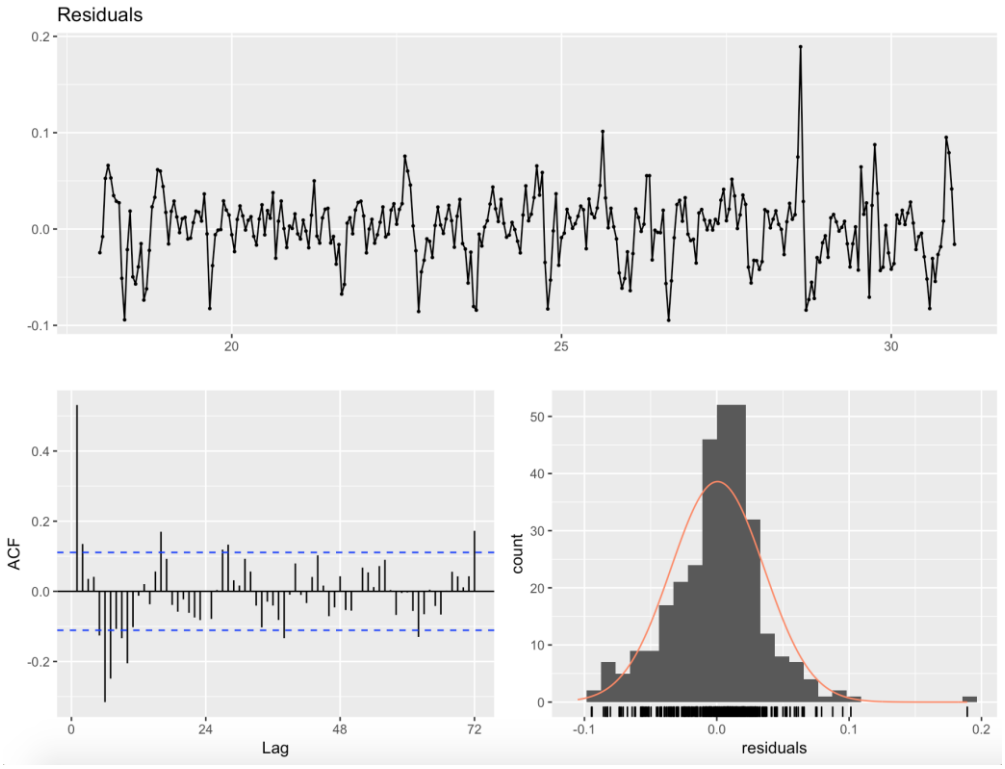


Figure 11: Residuals of the ETS method on the BoxCox transformed Humedad_Comedor_Sensor

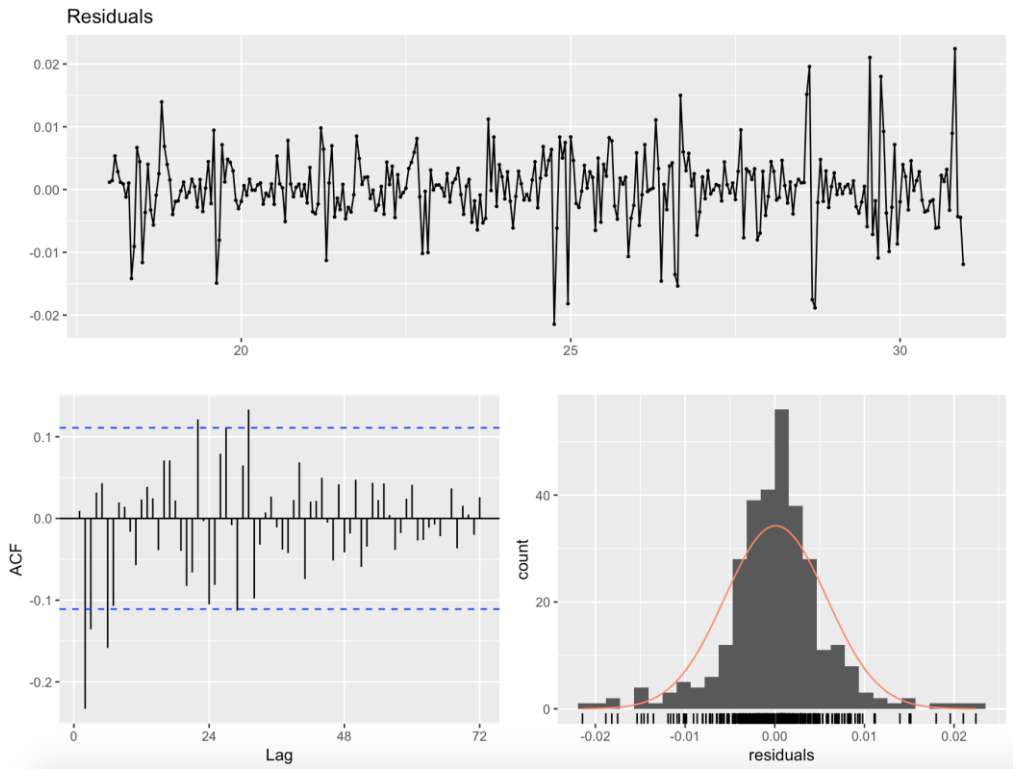


Figure 12: Residuals from ARIMA(2,1,1)(1,0,1)[24] of Humedad_Comedor_Sensor

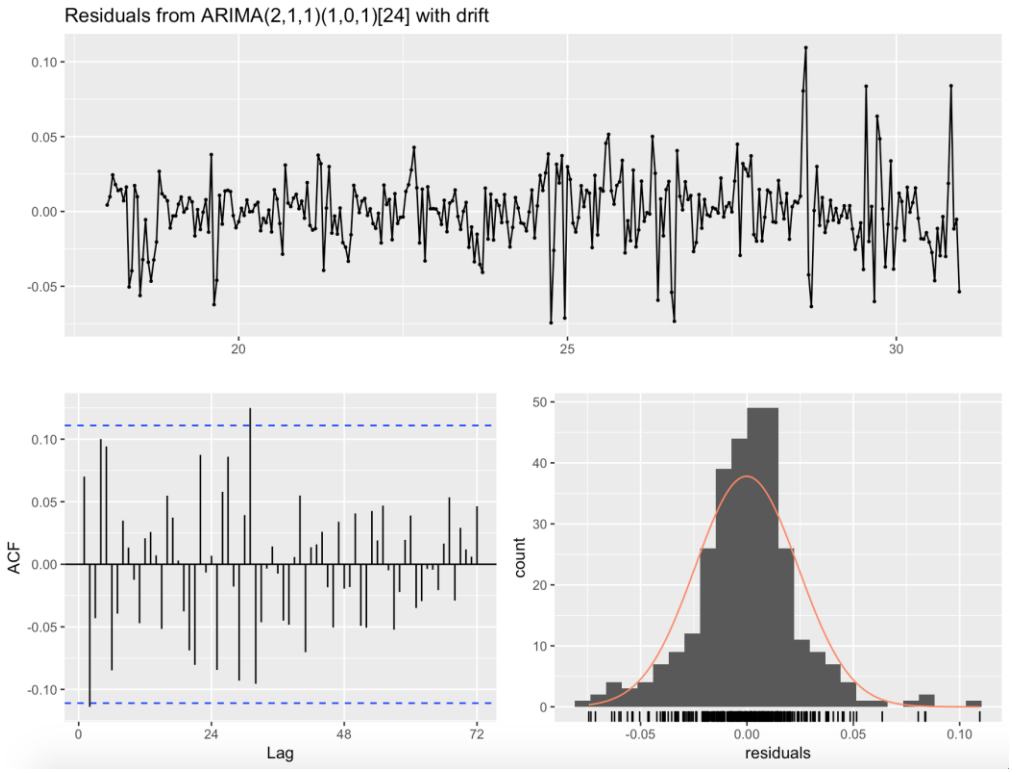


Figure 13: ggPacf of the residuals resulting from ARIMA(2,1,1)(1,0,1)[24] for Humedad_Comedor_Sensor.

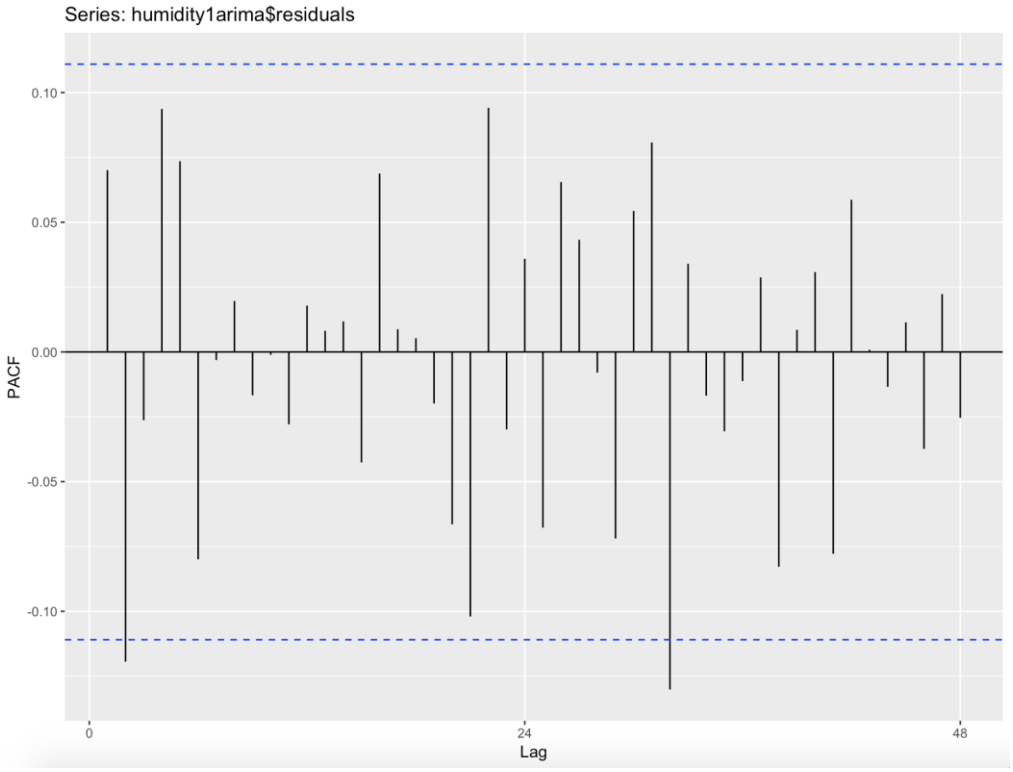


Figure 14: Unitroot test of the selected ARIMA(2,1,1)(1,0,1)[24] method for Humedad_Comedor_Sensor

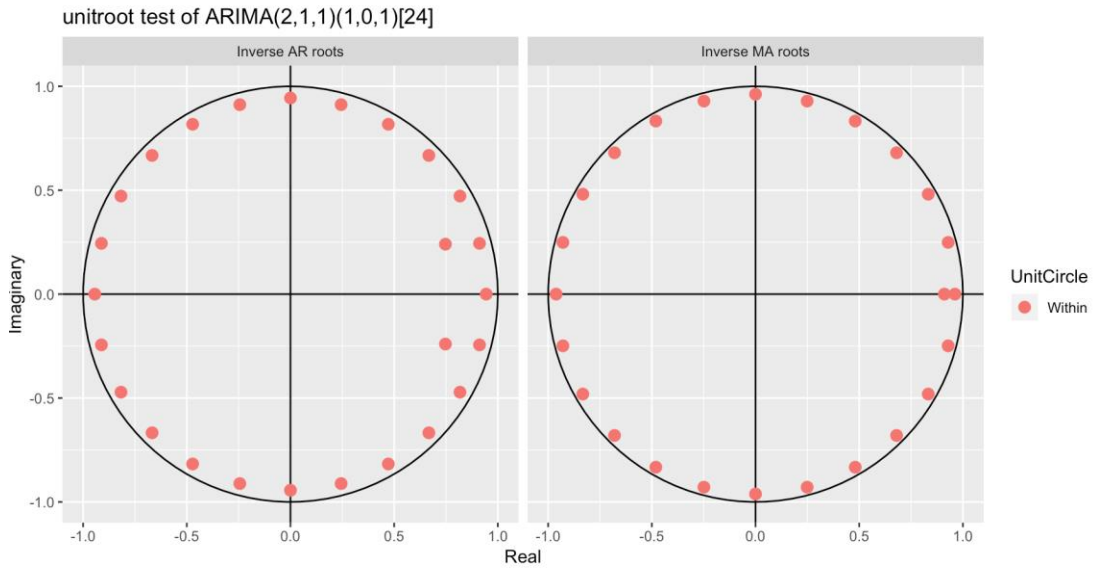


Figure 15: 24 hour forecast of Humedad_Comedor_Sensor using the ARIMA(2,1,1)(1,0,1)[24] with prediction intervals.

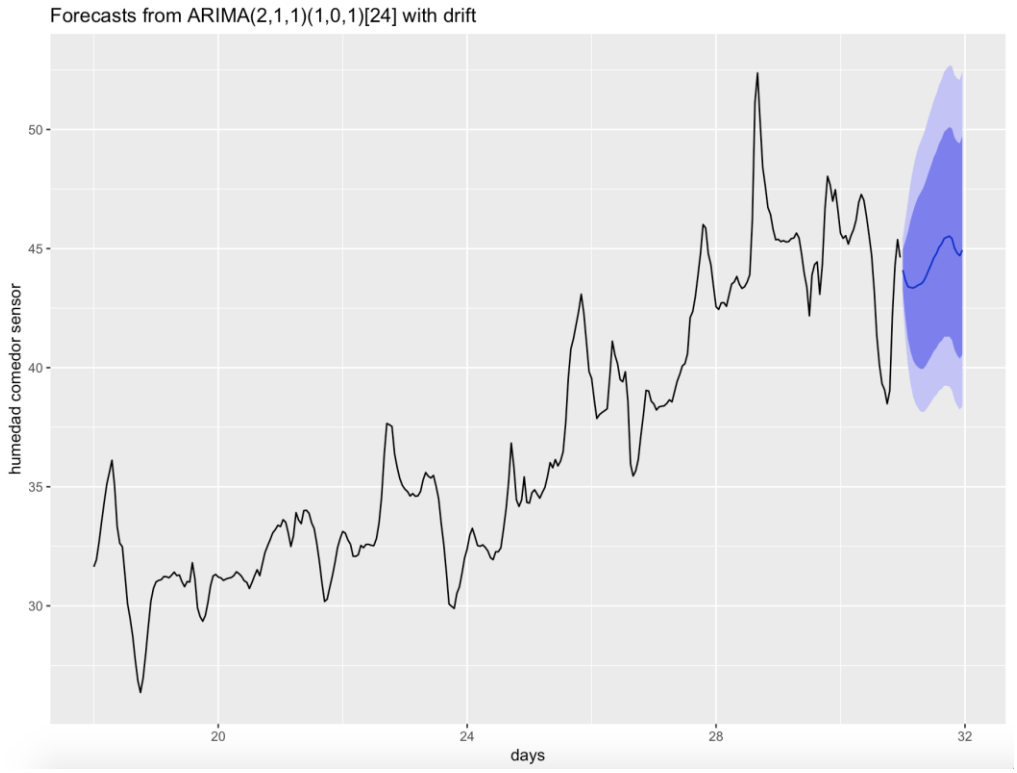


Figure 16: 1 hour forecast of Humedad_Comedor_Sensor with the selected ARIMA(2,1,1)(1,0,1)[24]

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
1	44.09558	43.23301	44.96414	42.7847	45.4324

"Lo 80" indicates the lower bound of the 80% prediction interval.
"Hi 80" indicates the higher bound of the 80% prediction interval.
"Lo 95" indicates the lower bound of the 95% prediction interval.
"Hi 95" indicated the higher bound of the 95% prediction interval.

Figure 17: 24 hours forecast of Humedad_Comedor_Sensor with the selected ARIMA(2,1,1)(1,0,1)[24]

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
1	44.09558	43.23301	44.96414	42.78470	45.43240
2	43.68118	42.09470	45.31071	41.27772	46.19716
3	43.40597	41.21036	45.69426	40.09014	46.95036
4	43.36605	40.69054	46.18524	39.33581	47.74419
5	43.34769	40.31947	46.56411	38.79490	48.35254
6	43.39603	40.11580	46.89962	38.47108	48.85536
7	43.47136	40.01546	47.17668	38.28751	49.25059
8	43.51440	39.93950	47.35724	38.15549	49.51207
9	43.62076	39.95787	47.56519	38.13235	49.77976
10	43.83820	40.10088	47.86781	38.24000	50.13221
11	44.09950	40.29701	48.20329	38.40502	50.51092
12	44.34394	40.48419	48.51286	38.56485	50.85845
13	44.61558	40.69931	48.84861	38.75291	51.23148
14	44.79160	40.82666	49.08034	38.85714	51.49582
15	45.05484	41.03341	49.40782	39.03692	51.86073
16	45.19768	41.12746	49.60692	39.10792	52.09291
17	45.42503	41.29727	49.90019	39.25040	52.42477
18	45.47654	41.30344	50.00482	39.23546	52.56096
19	45.52258	41.30316	50.10526	39.21364	52.69379
20	45.41385	41.15990	50.03847	39.05481	52.65248
21	45.01241	40.74809	49.65311	38.63955	52.27818
22	44.81039	40.51951	49.48464	38.39943	52.13059
23	44.70228	40.37740	49.41814	38.24210	52.08949
24	44.93400	40.54647	49.72238	38.38168	52.43652

"Lo 80" indicates the lower bound of the 80% prediction interval.
"Hi 80" indicates the higher bound of the 80% prediction interval.
"Lo 95" indicates the lower bound of the 95% prediction interval.
"Hi 95" indicated the higher bound of the 95% prediction interval.

Figure 18: Seasonal plot of Humedad_Exterior_Sensor.

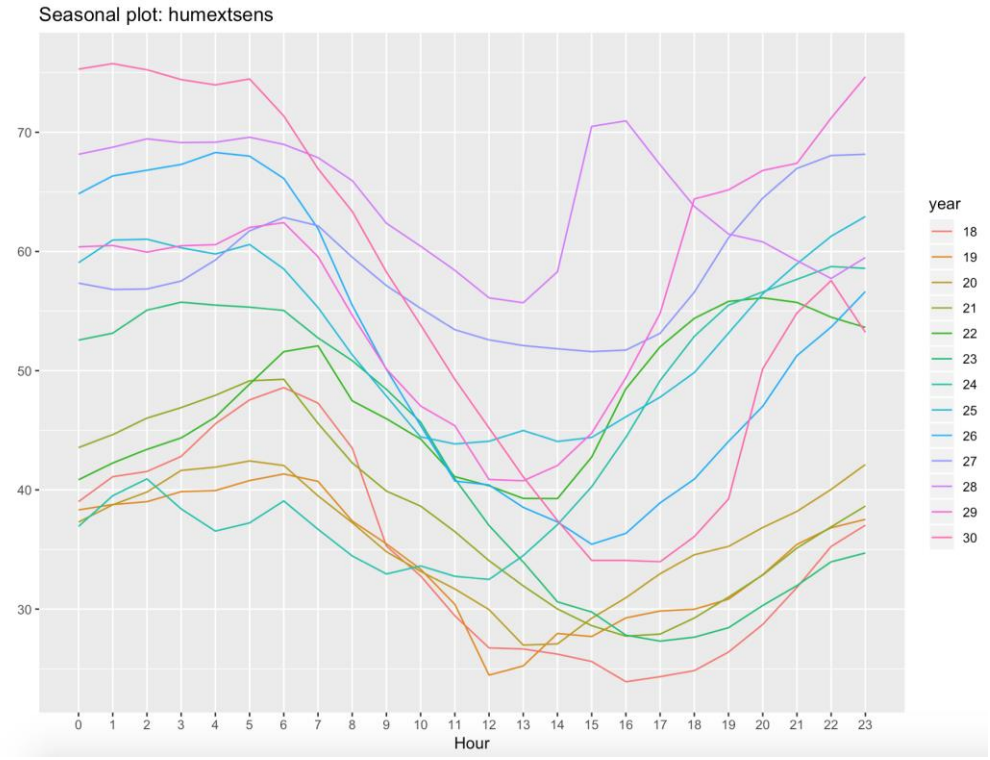


Figure 19: Subseasonal plot of Humedad_Exterior_Sensor

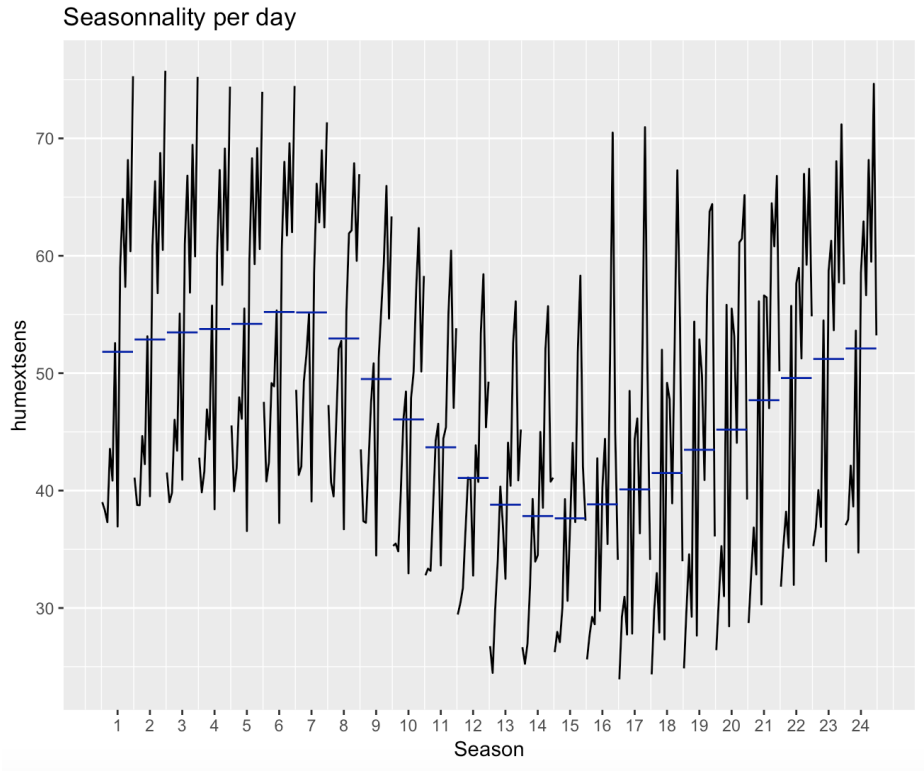


Figure 20: STL decomposition of Humedad_Exterior_Sensor

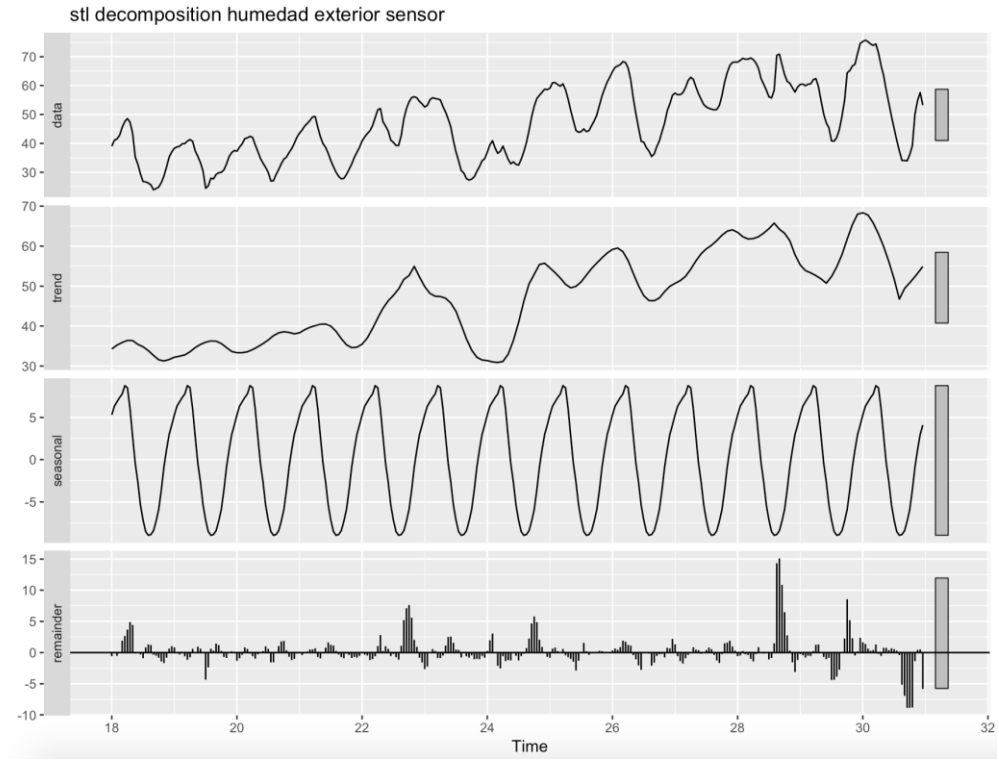


Figure 21: BoxCox transformed Humedad_Exterior_Sensor

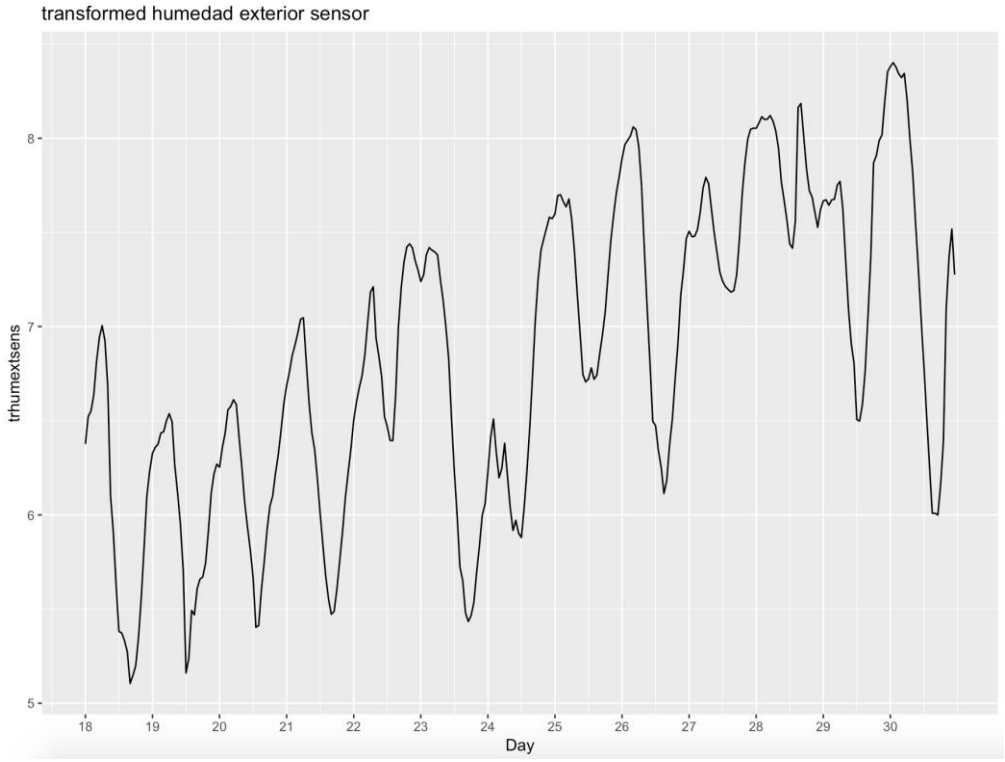


Figure 22: Residuals from the Holt-Winters' method on the BoxCox transformed Humedad_Exterior_Sensor.

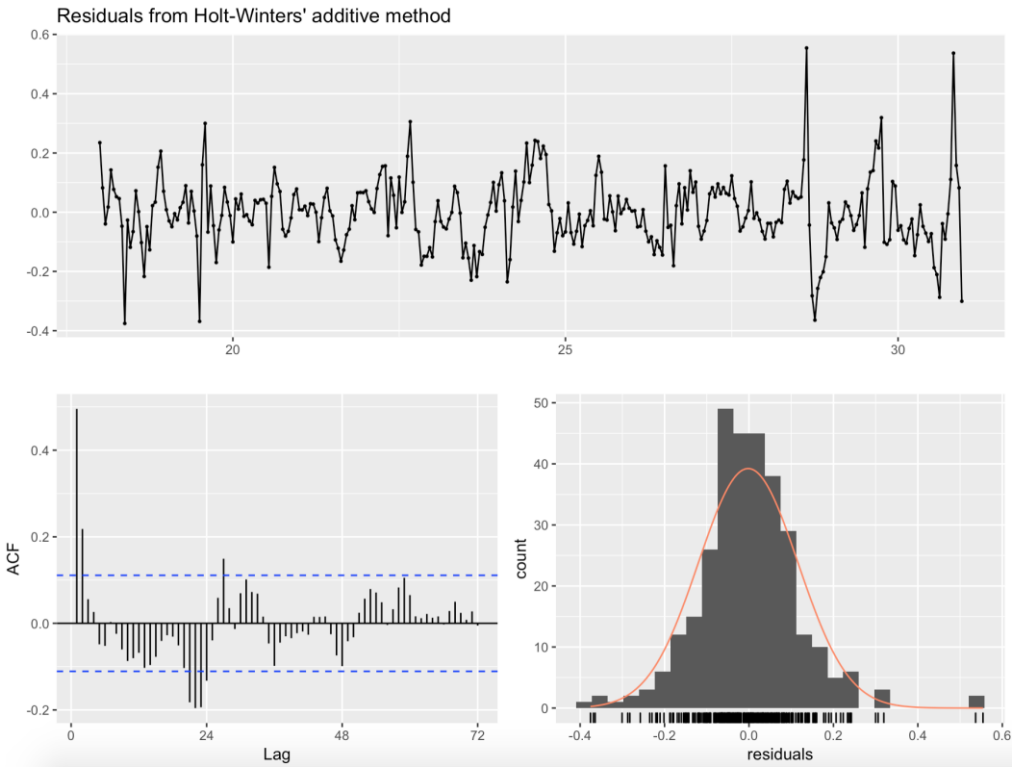


Figure 23: Residuals from ETS (M,Ad,N) on the BoxCox transformed Humedad_Exterior_Sensor

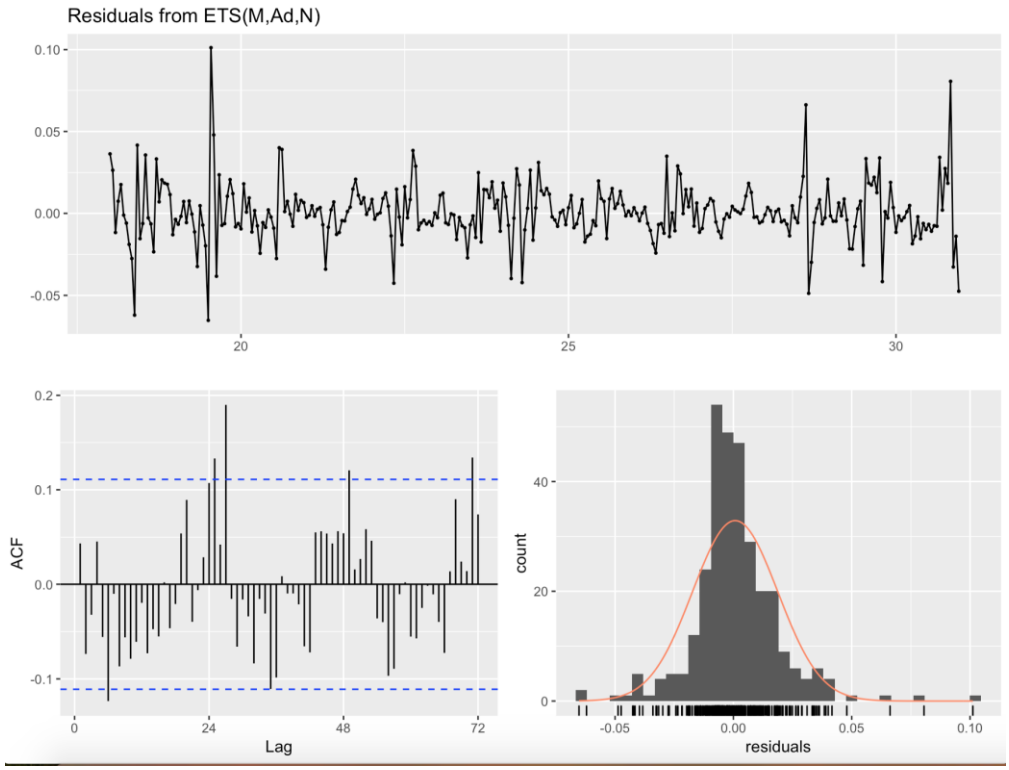


Figure 24: Residuals from ARIMA(2,0,2)(1,1,0)[24] of Humedad_Exterior_Sensor

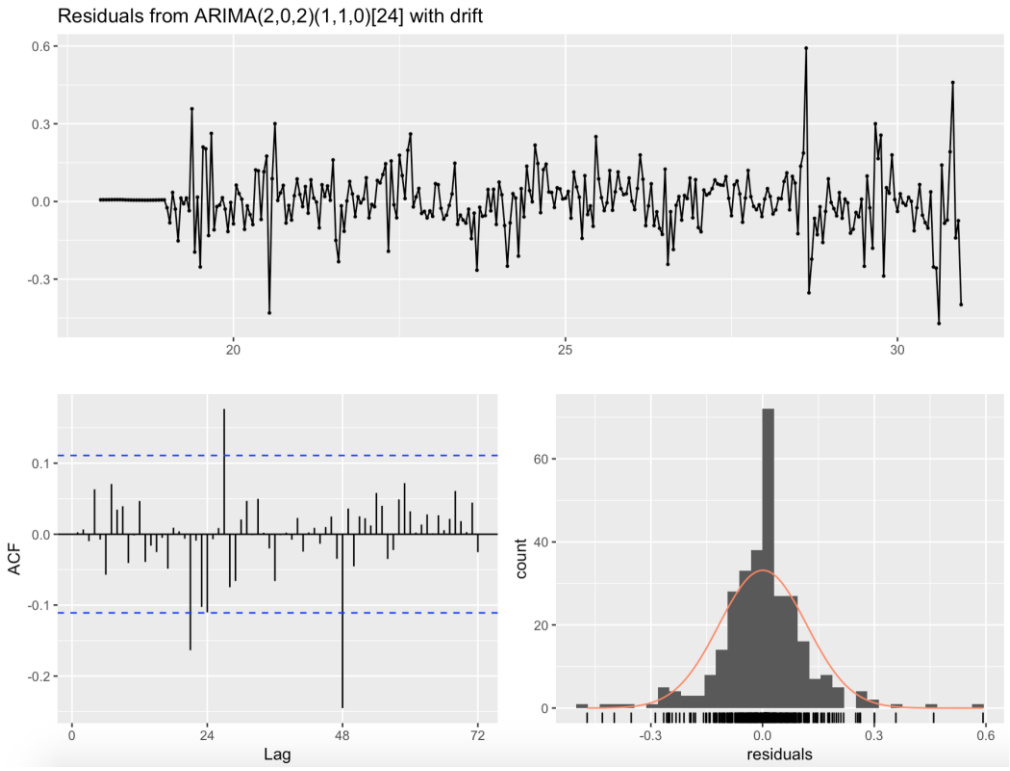


Figure 25: ggPACF of the ARIMA(2,0,2)(1,1,0)[24] model for Humedad_Exterior_Sensor

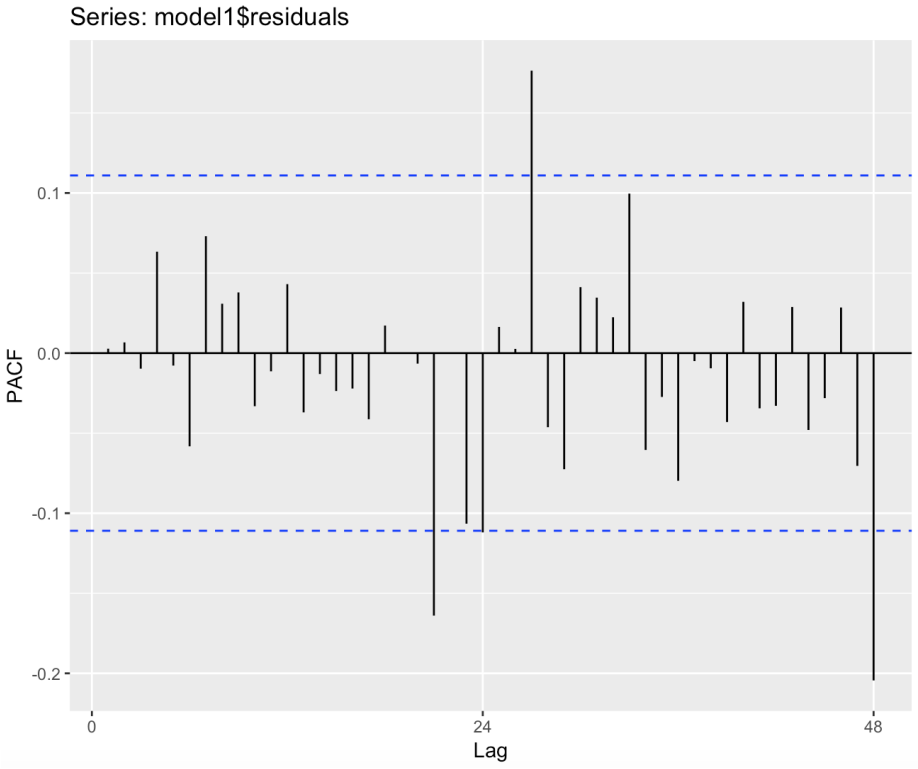


Figure 26: Unitroot test of ARIMA(2,0,2)(1,1,0)[24] model for Humedad_Exterior_Sensor

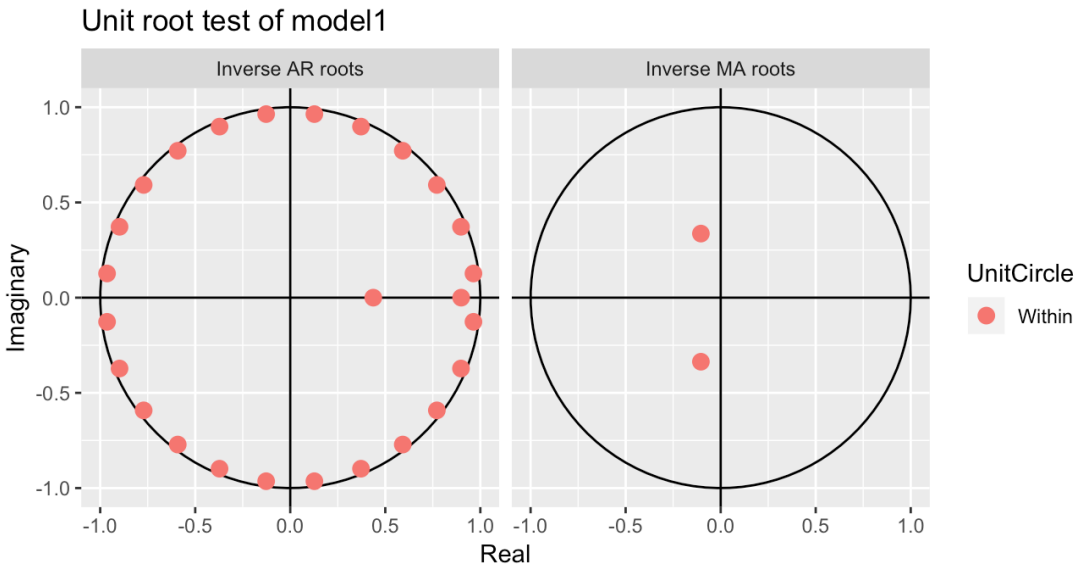


Figure 27: 24 hours forecast of Humedad_Exterior_Sensor using ARIMA(2,0,2)(1,1,0)[24] with the 80% and 95% prediction intervals computed in blue and light blue respectively.

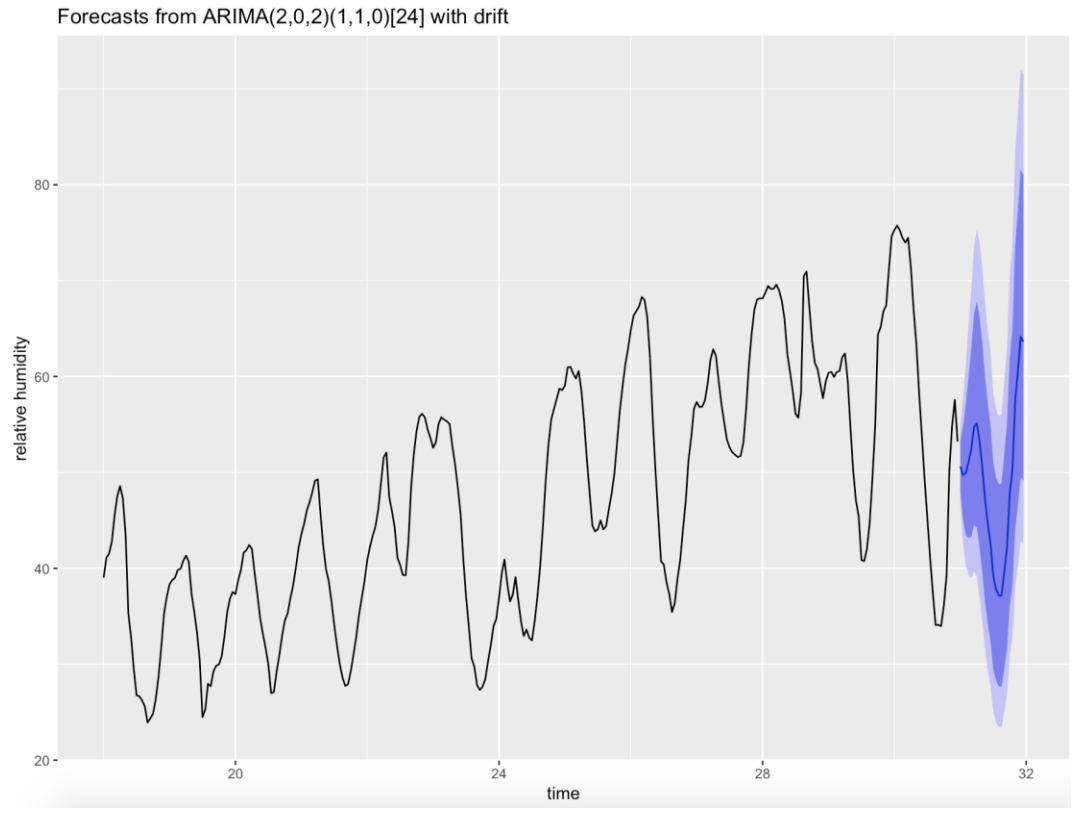


Figure 28: 1-hour forecast of Humedad_Exterior_Sensor using ARIMA(2,0,2)(1,1,0)[24]

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
1	50.62346	47.99667	53.35238	46.64666	54.83913

"Lo 80" indicates the lower bound of the 80% prediction interval.
"Hi 80" indicates the higher bound of the 80% prediction interval.
"Lo 95" indicates the lower bound of the 95% prediction interval.
"Hi 95" indicated the higher bound of the 95% prediction interval.

Figure 29: 24 hours forecast of Humedad_Exterior_Sensor using ARIMA(2,0,2)(1,1,0)[24]

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
1	50.62346	47.99667	53.35238	46.64666	54.83913
2	49.74448	45.05042	54.78138	42.69929	57.59164
3	49.91414	43.43339	57.06476	40.26018	61.13515
4	51.09282	43.14410	60.04691	39.31847	65.21920
5	52.42014	43.29355	62.85866	38.95684	68.95284
6	54.75693	44.53700	66.56852	39.72358	73.51431
7	55.11832	44.24339	67.80003	39.16059	75.30360
8	53.25391	42.20126	66.25539	37.07393	73.99400
9	50.55456	39.58472	63.56337	34.53135	71.34900
10	47.22253	36.54636	59.98285	31.66183	67.66038
11	44.65882	34.22991	57.20457	29.48553	64.78594
12	42.60165	32.39008	54.95253	27.76666	62.44322
13	39.21178	29.50739	51.03005	25.14022	58.23062
14	37.84041	28.30822	49.49452	24.03346	56.61357
15	37.18113	27.70807	48.79261	23.46951	55.89767
16	37.16031	27.63862	48.84650	23.38323	56.00340
17	39.58849	29.55953	51.86550	25.06707	59.37136
18	42.19807	31.64004	55.08687	26.89887	62.95225
19	47.77672	36.15653	61.87490	30.90971	70.44301
20	50.30797	38.20216	64.96195	32.72507	73.85434
21	57.51268	44.10773	73.63225	38.00739	83.37074
22	60.61610	46.65340	77.36645	40.28594	87.46985
23	64.13943	49.55738	81.58728	42.89227	92.09295
24	63.63358	49.12328	81.00581	42.49440	91.47011

"Lo 80" indicates the lower bound of the 80% prediction interval.
"Hi 80" indicates the higher bound of the 80% prediction interval.
"Lo 95" indicates the lower bound of the 95% prediction interval.
"Hi 95" indicated the higher bound of the 95% prediction interval.

Figure 30 : Seasonal plot of Temperature_Comedor_Sensor

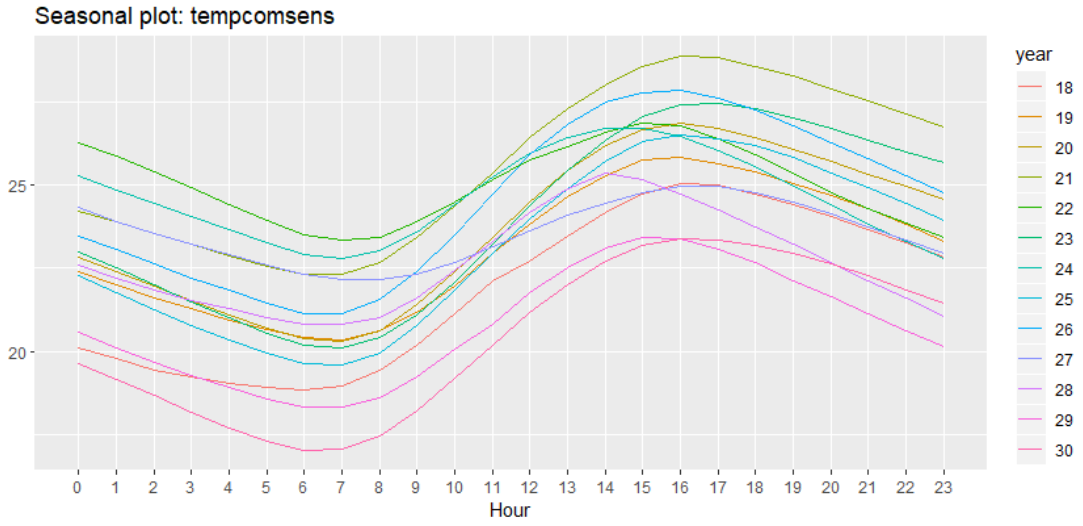


Figure 31 : Subseasonal plot of Temperature_Comedor_Sensor

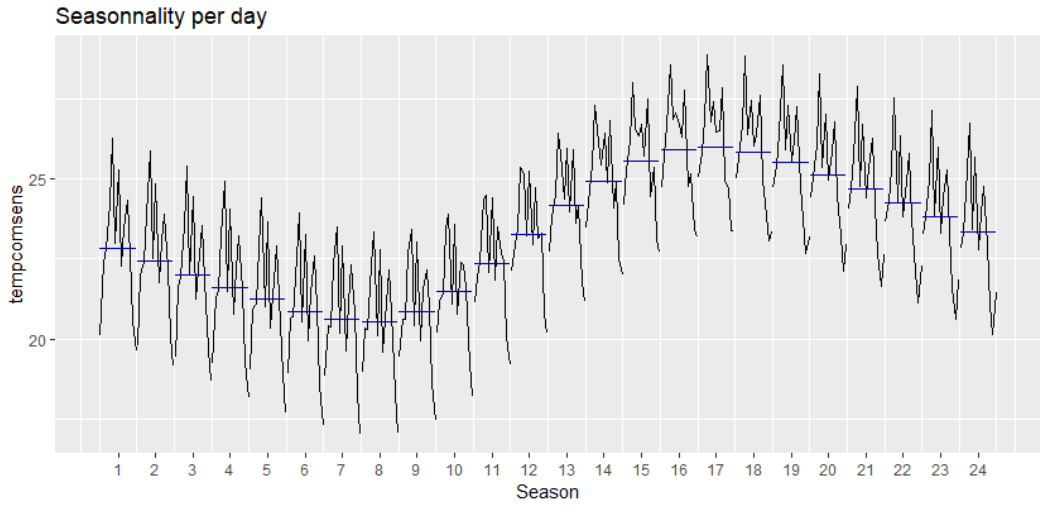


Figure 32 : ACF of Tempereature_Comedor_Sensor

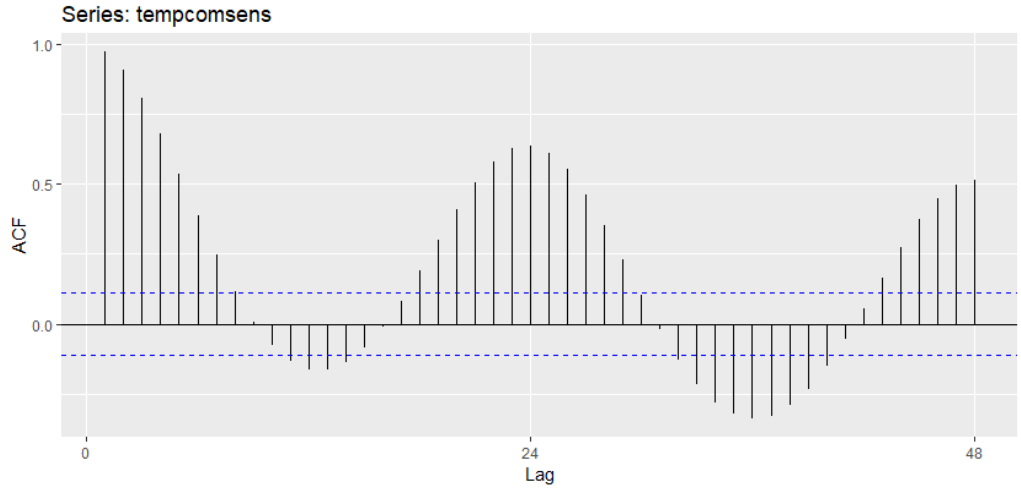


Figure 33 : STL decomposition of Temperature_Comedor_Sensor

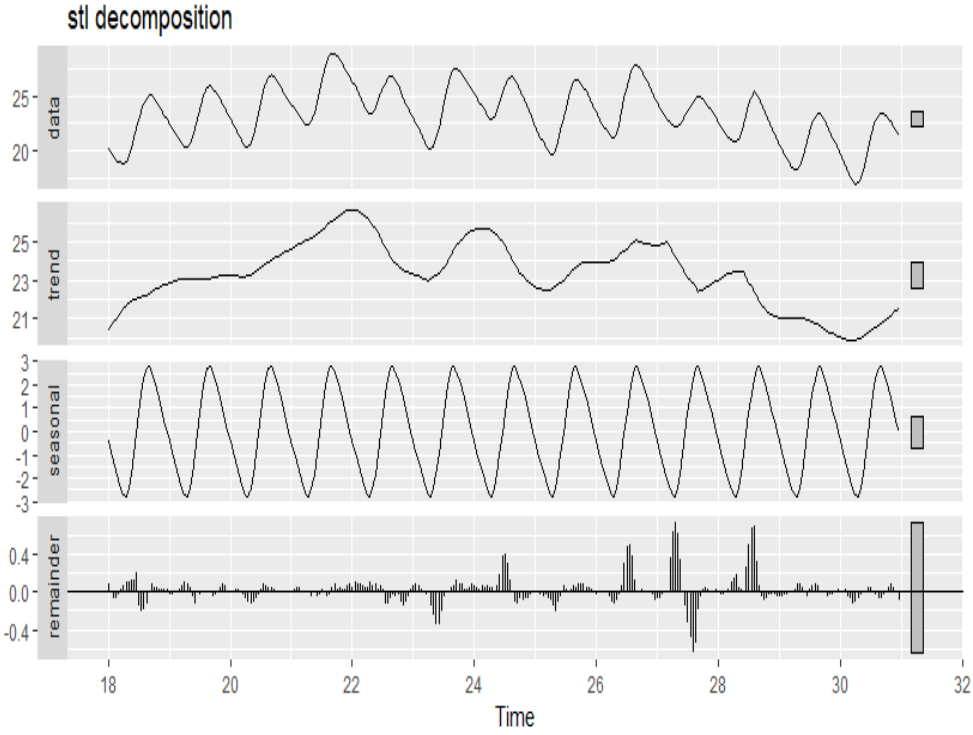


Figure 34 : Residuals for ARIMA(2,1,0)(2,1,0)[24] for Temperature_Comedor_Sensor

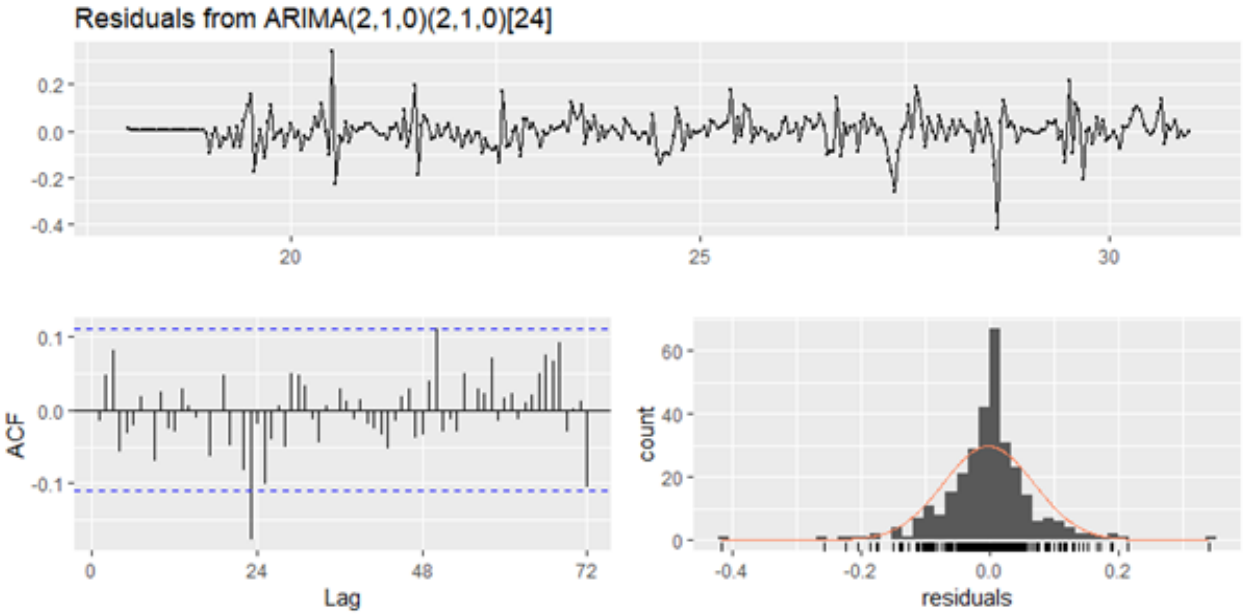


Figure 35 : Residuals for ETS(A, Ad, N) for Temperature_Comedor_Sensor

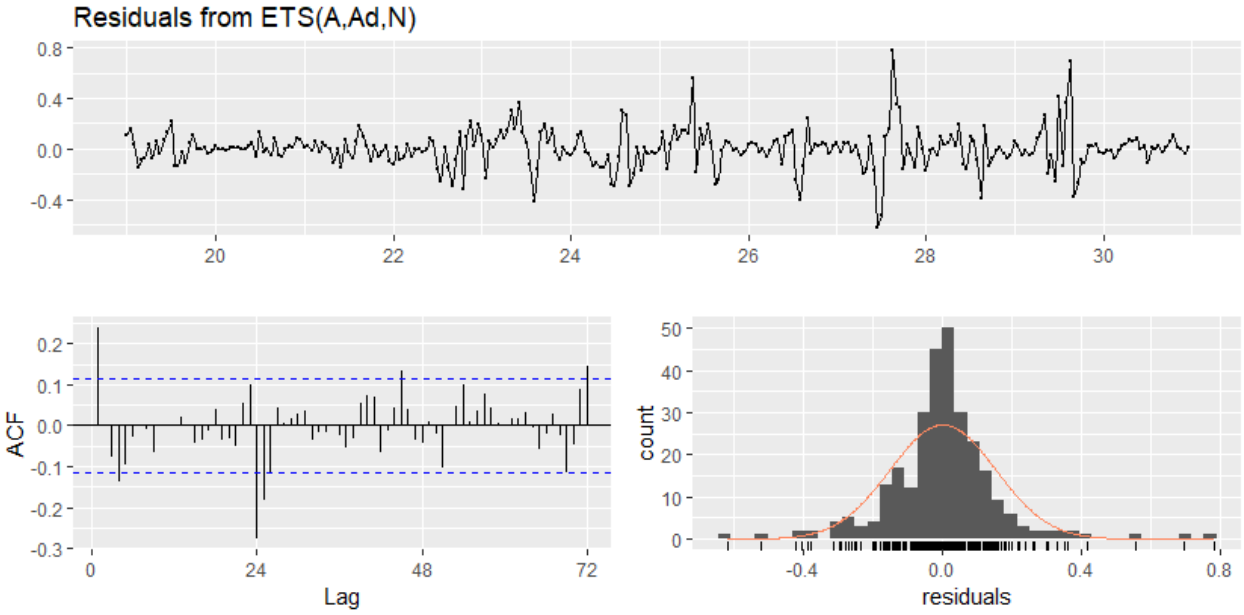


Figure 36 : Unitroot test of ARIMA(2,1,0)(2,1,0)[24] model for Temperature_Comedor_Sensor

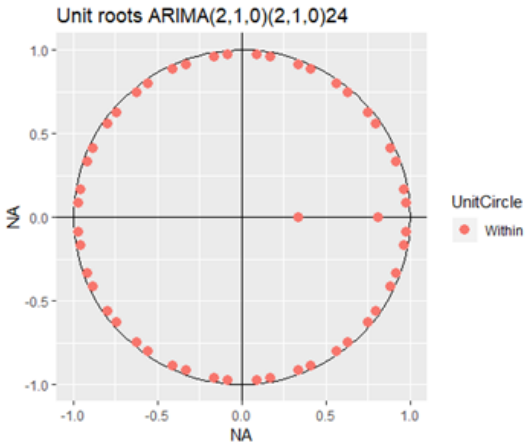


Figure 37 : Short range forecast of Temperature_Comedor_Sensor (h=1)

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
31	20.99984	20.90548	21.09421	20.85552	21.14417

"Lo 80" indicates the lower bound of the 80% prediction interval.
"Hi 80" indicates the higher bound of the 80% prediction interval.
"Lo 95" indicates the lower bound of the 95% prediction interval.
"Hi 95" indicated the higher bound of the 95% prediction interval.

Figure 38 : 24 hours forecast of Temperature_Comedor_Sensor

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
31.00000	20.99984	20.90548	21.09421	20.85552	21.14417
31.04167	20.57623	20.35289	20.79956	20.23466	20.91779
31.08333	20.17386	19.79950	20.54823	19.60132	20.74641
31.12500	19.77359	19.23783	20.30936	18.95421	20.59298
31.16667	19.41216	18.71157	20.11274	18.34071	20.48360
31.20833	19.07835	18.21363	19.94307	17.75588	20.40083
31.25000	18.84124	17.81544	19.86704	17.27242	20.41007
31.29167	18.86455	17.68207	20.04702	17.05610	20.67299
31.33333	19.17712	17.84306	20.51117	17.13685	21.21738
31.37500	19.85085	18.37062	21.33108	17.58704	22.11467
31.41667	20.74445	19.12351	22.36540	18.26543	23.22347
31.45833	21.63781	19.88151	23.39411	18.95178	24.32383
31.50000	22.57916	20.69268	24.46565	19.69403	25.46429
31.54167	23.35839	21.34664	25.37014	20.28169	26.43510
31.58333	23.94883	21.81647	26.08119	20.68767	27.20999
31.62500	24.18537	21.93678	26.43396	20.74645	27.62429
31.66667	24.11721	21.75649	26.47794	20.50680	27.72763
31.70833	23.86718	21.39815	26.33622	20.09112	27.64324
31.75000	23.52682	20.95305	26.10059	19.59058	27.46306
31.79167	23.11562	20.44045	25.79080	19.02430	27.20695
31.83333	22.68969	19.91622	25.46316	18.44803	26.93135
31.87500	22.22112	19.35225	25.09000	17.83355	26.60869
31.91667	21.75037	18.78879	24.71194	17.22103	26.27971
31.95833	21.26797	18.21622	24.31972	16.60072	25.93522

"Lo 80" indicates the lower bound of the 80% prediction interval.
"Hi 80" indicates the higher bound of the 80% prediction interval.
"Lo 95" indicates the lower bound of the 95% prediction interval.
"Hi 95" indicated the higher bound of the 95% prediction interval.

Figure 39 :Forecast of Temperature_Comedor_Sensor for 24 hours for ARIMA(2,1,0)(2,1,0)[24] method with the 80% and 95% prediction intervals

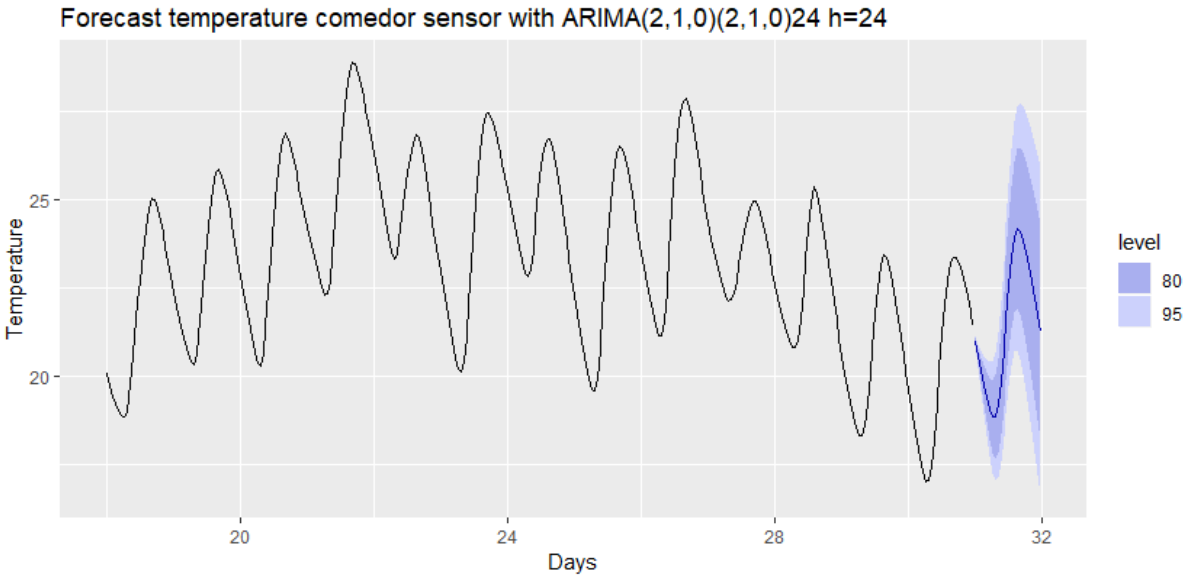


Figure 40 : seasonal plot of C02_Comedor_Sensor

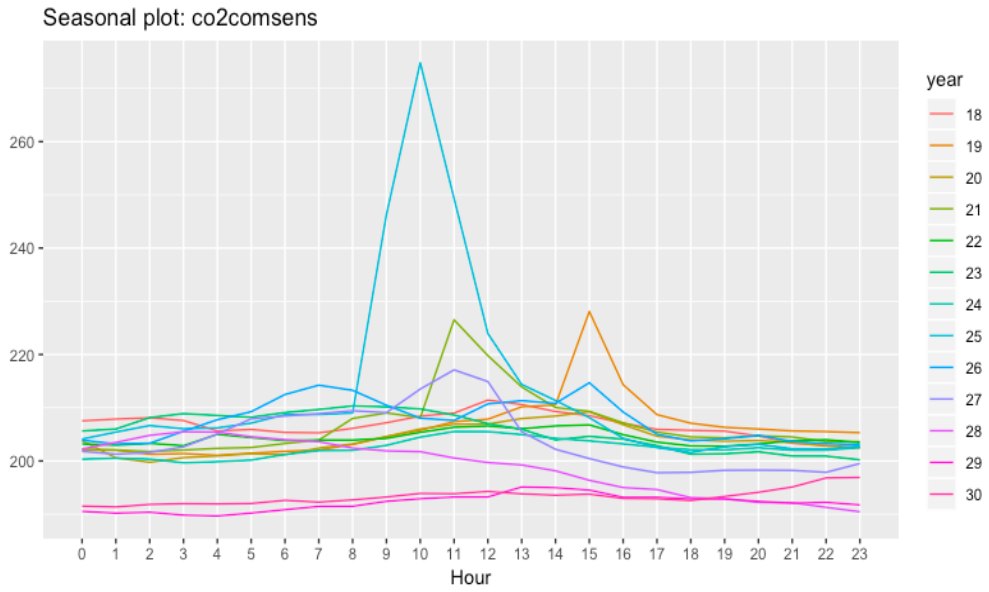


Figure 41 :Subseasonal plot of CO2_Comedor_Sensor

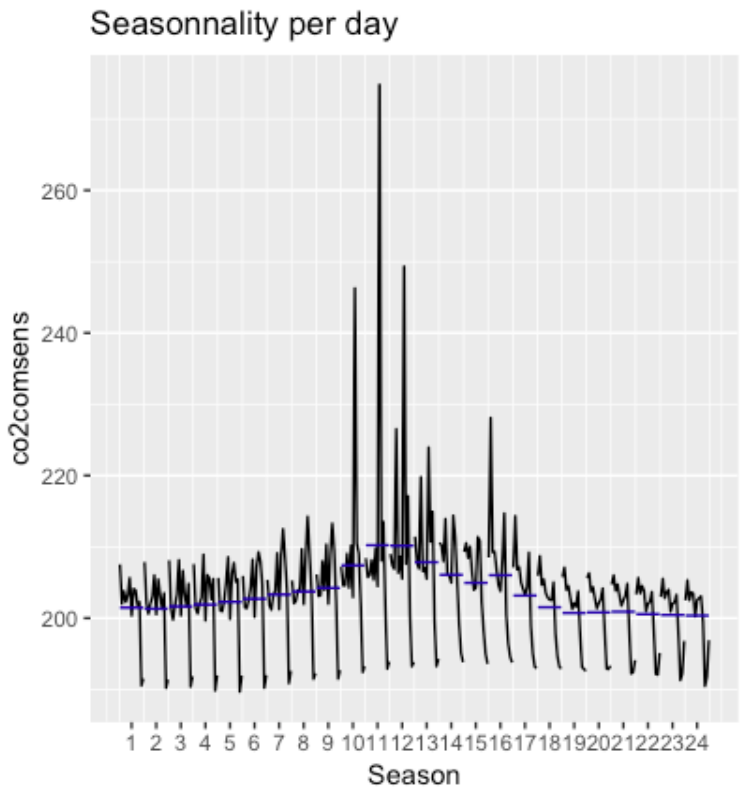


Figure 42 : ACF of CO2_Comedor_Sensor

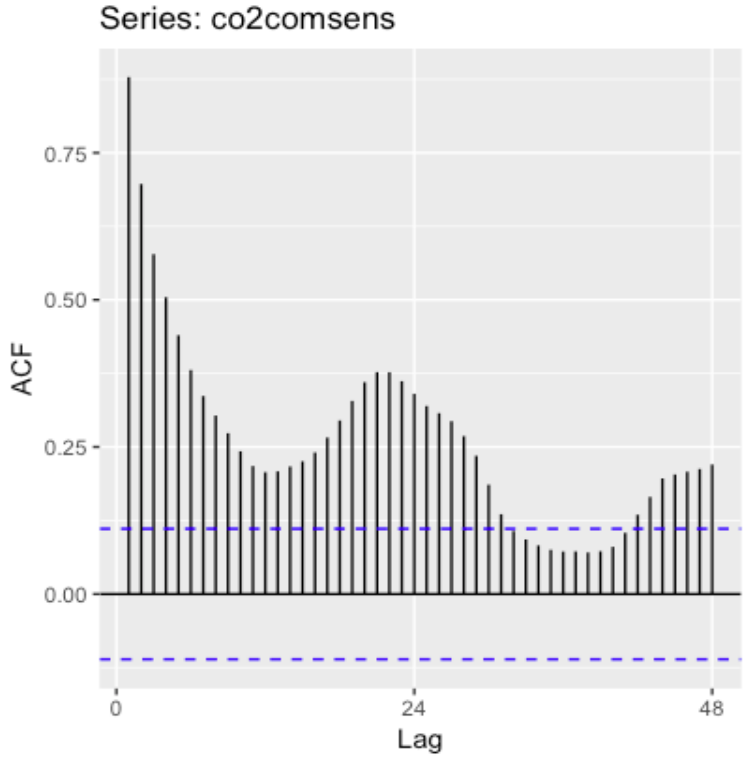


Figure 43 : STL decomposition of C02_Comedor_Sensor

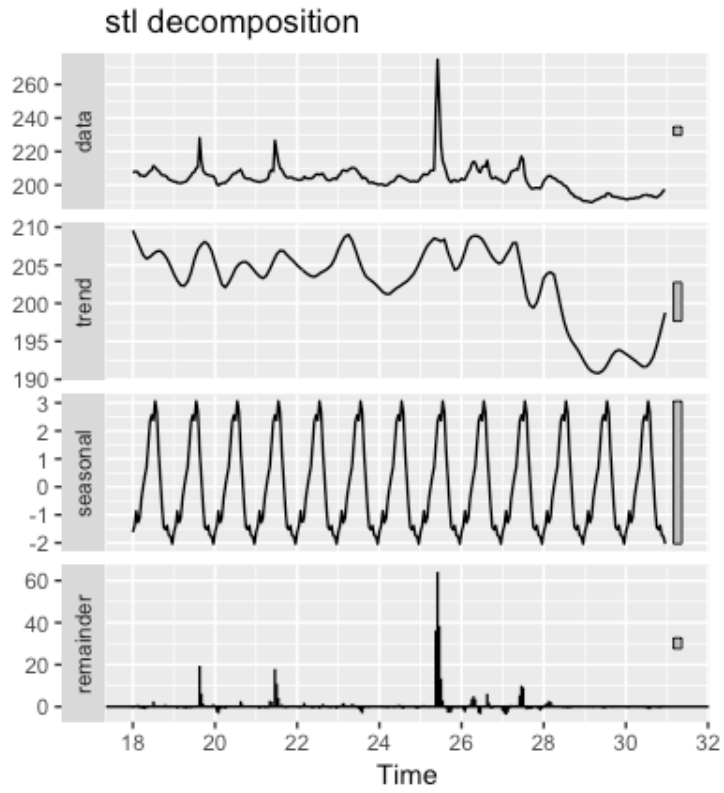


Figure 44 : ARIMA(1,1,2)(2,0,0)[24] residuals of CO2_Comedor_Sensor

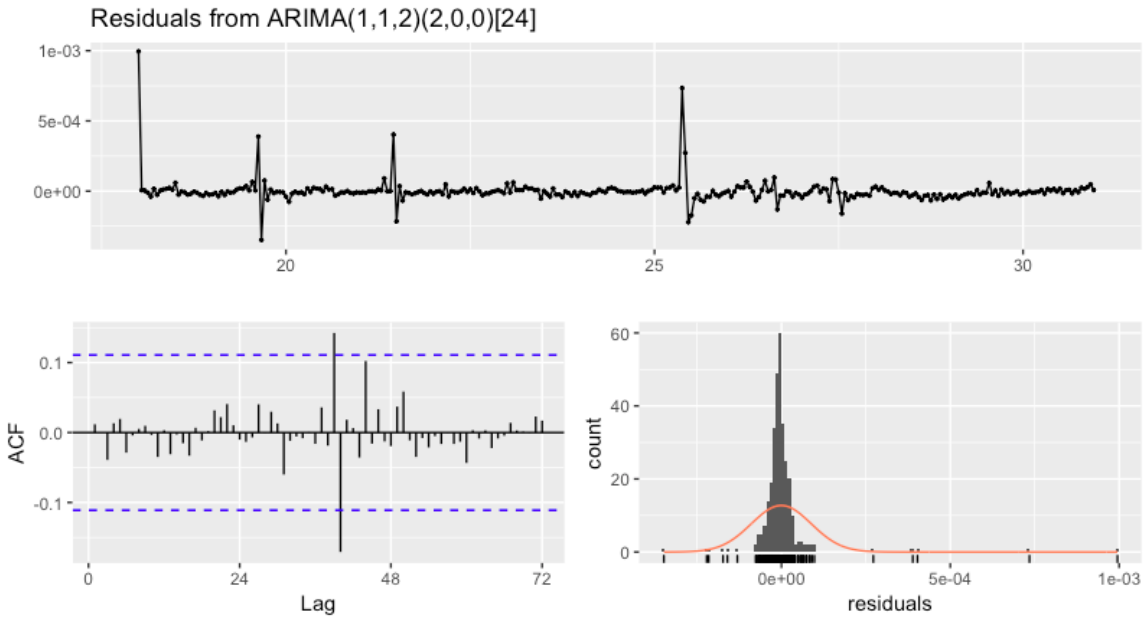


Figure 45 : ETS(A,N,N) residuals for CO2_Comedor_Sensor

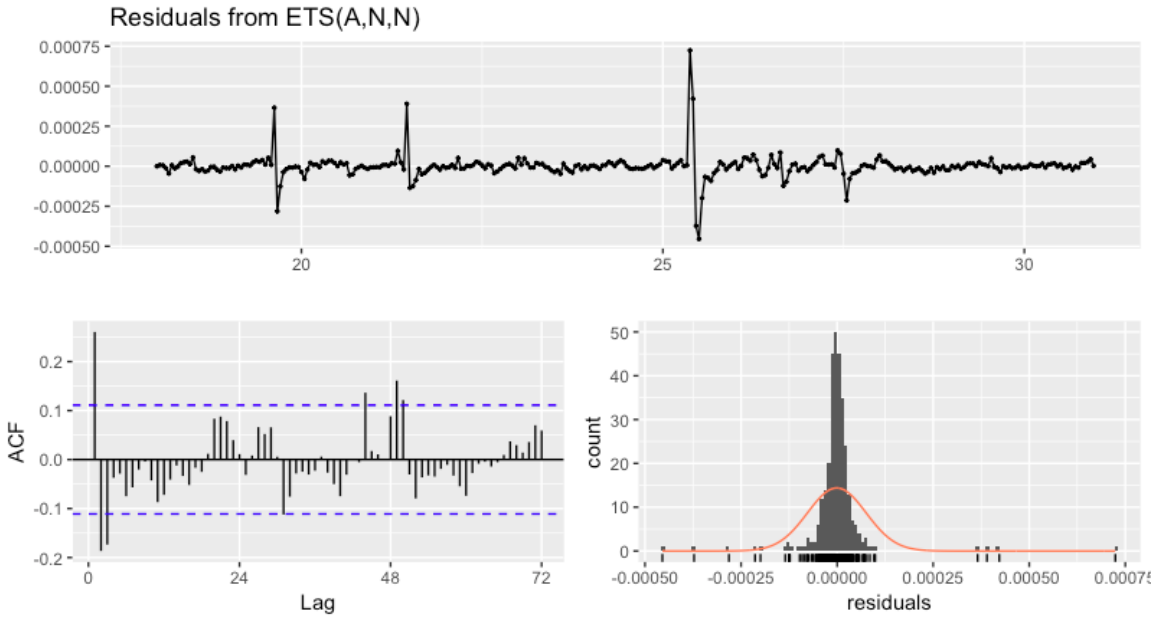


Figure 46 : Unitroot test of CO2_Comedor_Sensor

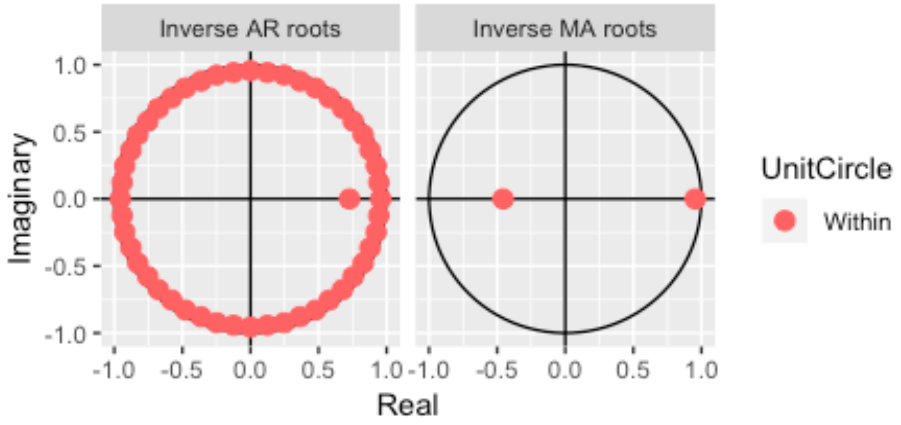


Figure 47 : 24 hour forecast of CO2_Comedor_Sensor using ARIMA(1,1,2)(2,0,0)[24]

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
31.00000	196.1512	191.7829	200.5990	189.5776	203.0698
31.04167	195.3911	188.6419	202.5152	185.2823	206.5355
31.08333	194.9119	187.1128	203.2628	183.2589	208.0149
31.12500	194.4860	186.1168	203.5166	181.9983	208.6804
31.16667	194.1955	185.4780	203.6461	181.1993	209.0665
31.20833	194.0575	185.1039	203.7945	180.7169	209.3907
31.25000	194.0066	184.8811	203.9528	180.4157	209.6778
31.29167	193.9508	184.6976	204.0529	180.1740	209.8743
31.33333	193.8955	184.5422	204.1206	179.9731	210.0180
31.37500	193.9589	184.5120	204.2980	179.9003	210.2660
31.41667	193.9972	184.4710	204.4334	179.8233	210.4615
31.45833	193.9982	184.4044	204.5174	179.7261	210.5971
31.50000	193.9973	184.3418	204.5925	179.6356	210.7195
31.54167	194.1492	184.4210	204.8327	179.6818	211.0143
31.58333	194.1139	184.3343	204.8611	179.5718	211.0826
31.62500	194.0692	184.2413	204.8765	179.4571	211.1355
31.66667	193.8923	184.0309	204.7425	179.2319	211.0288
31.70833	193.8861	183.9755	204.7973	179.1544	211.1217
31.75000	193.8360	183.8815	204.8019	179.0407	211.1606
31.79167	193.8662	183.8606	204.8953	178.9968	211.2935
31.83333	193.8552	183.8033	204.9421	178.9186	211.3765
31.87500	193.8606	183.7612	205.0066	178.8552	211.4780
31.91667	193.9497	183.7947	205.1642	178.8635	211.6784
31.95833	193.9043	183.7077	205.1709	178.7580	211.7181

“Lo 80” indicates the lower bound of the 80% prediction interval.
“Hi 80” indicates the higher bound of the 80% prediction interval.
“Lo 95” indicates the lower bound of the 95% prediction interval.
“Hi 95” indicated the higher bound of the 95% prediction interval.

Figure 48 : seasonal plot of Lighting_Habitacion_Sensor

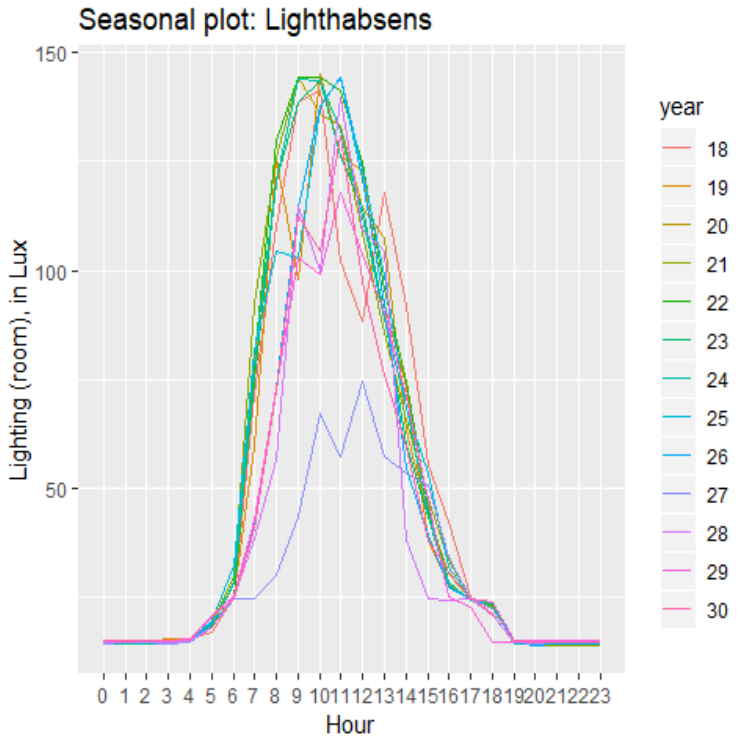


Figure 49 : Lagplots of Lighting_Habitacion_Sensor

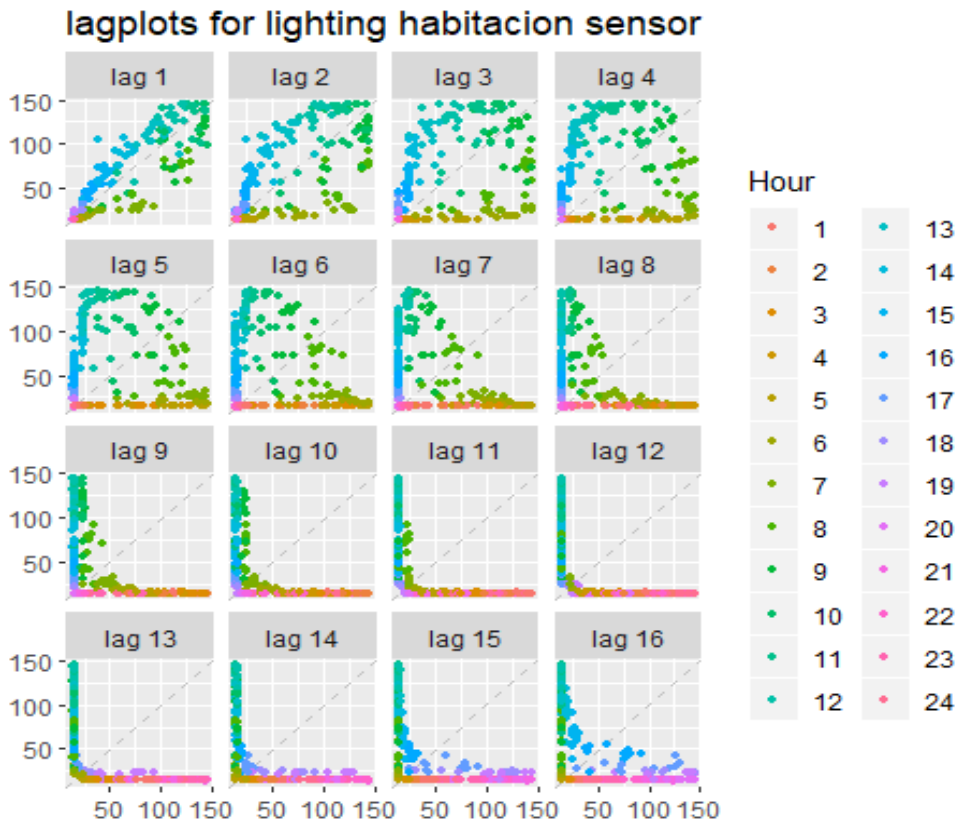


Figure 50 : Subseasonal plot of Lighting_Habitacion_Sensor

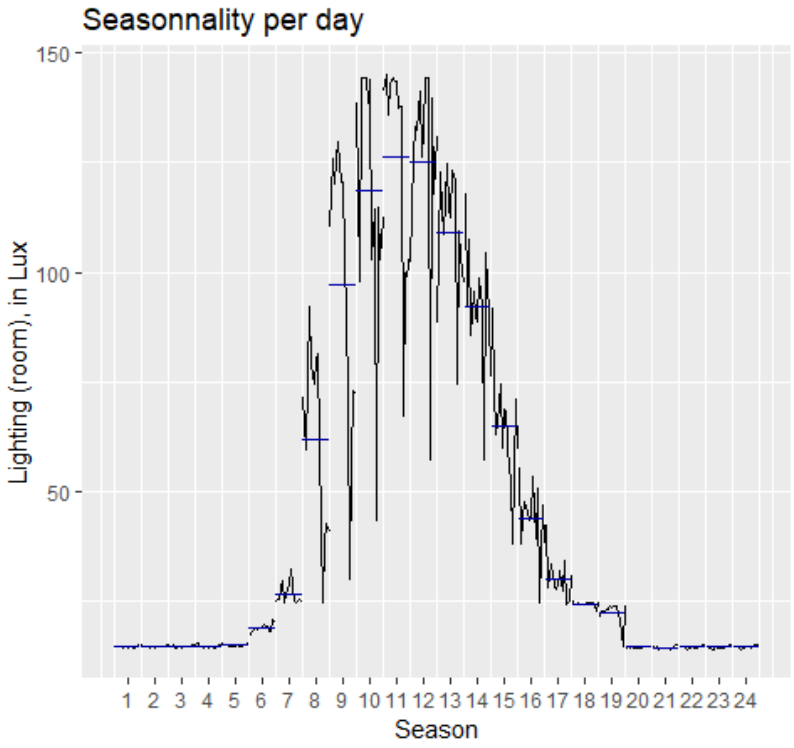


Figure 51 : ACF of Lighting_Habitacion_Sensor

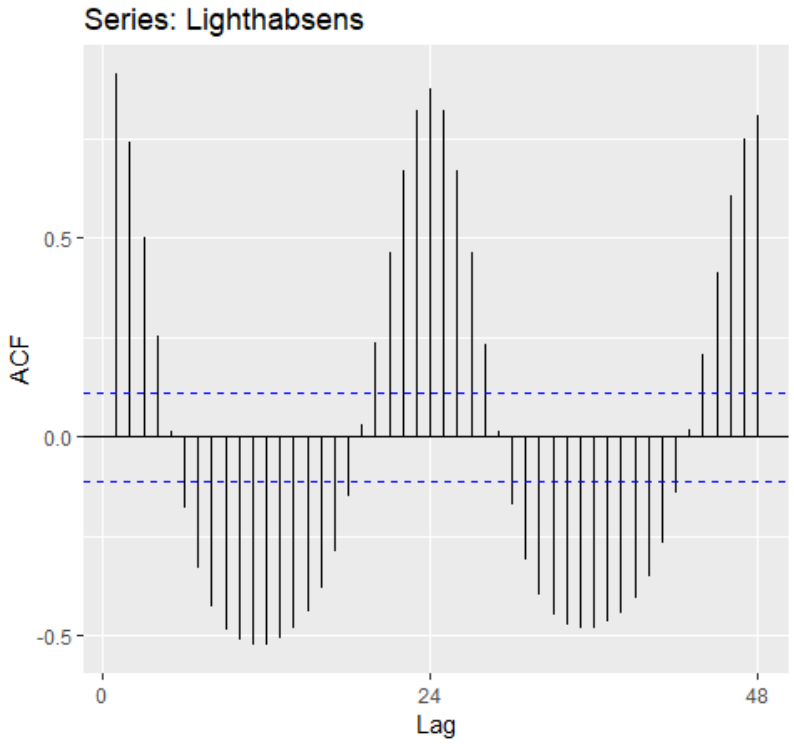


Figure 52 : STL decompositon of Lighting_Habitacion_Sensor

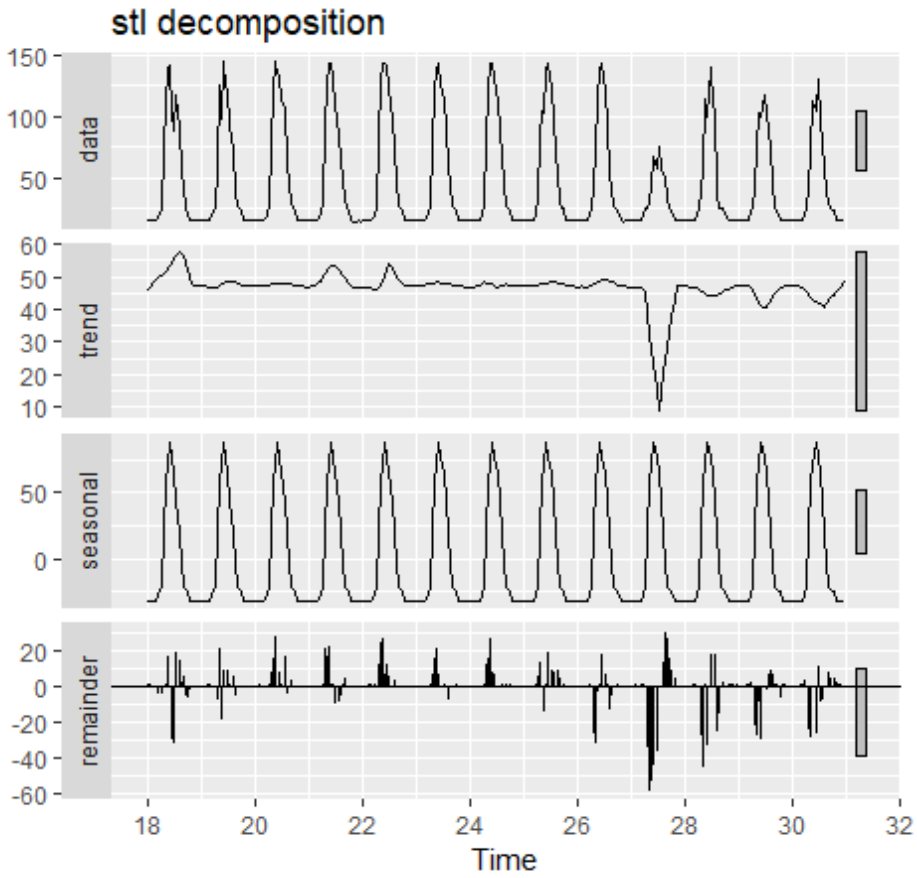


Figure 53 : BoxCox transformed Lighting_Habitacion_Sensor

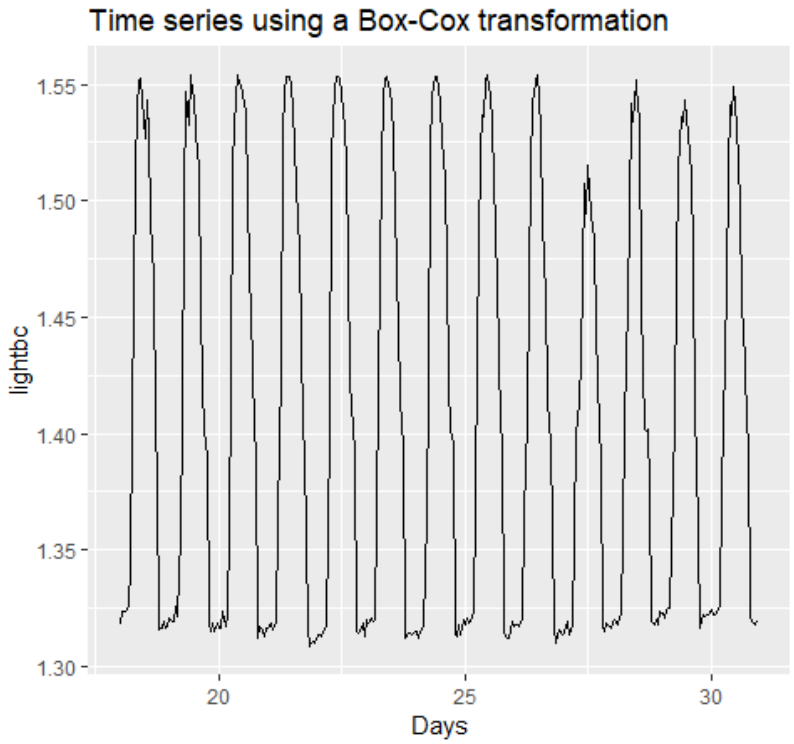


Figure 54 : Residuals of ARIMA(1,0,0)(1,1,2)[24] method for Lighting_Habitacion_Sensor

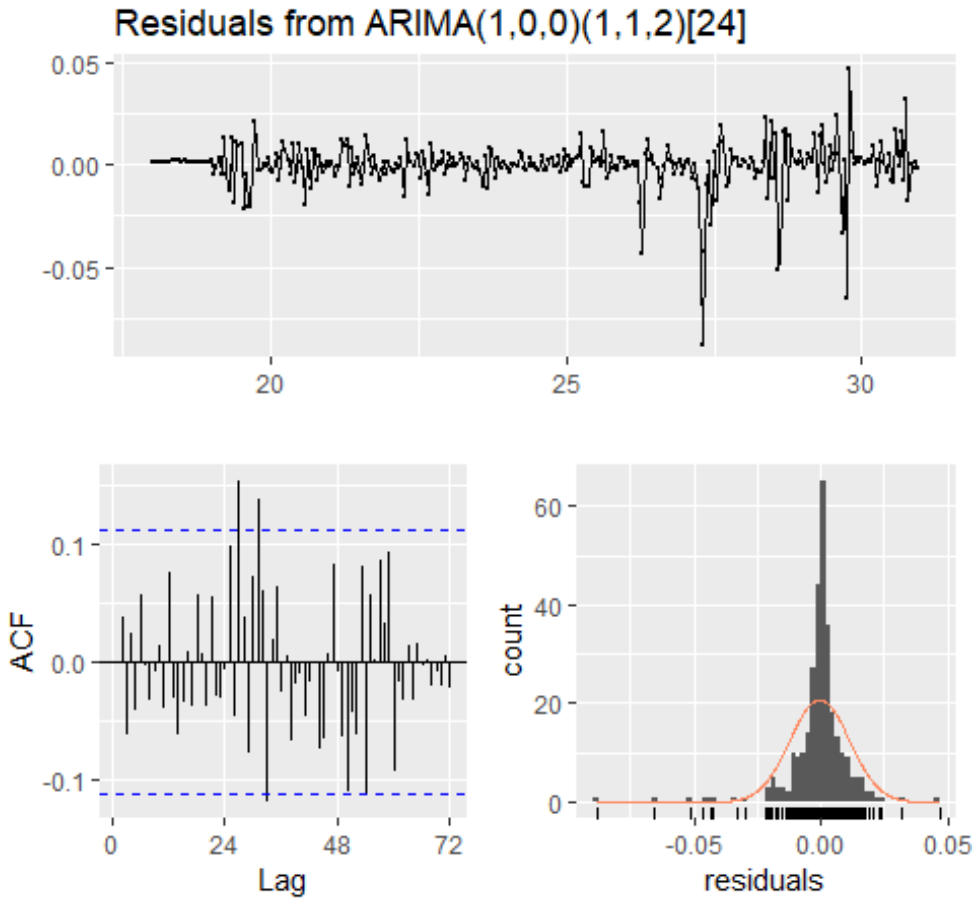


Figure 55: Unit root test for ARIMA(1,0,0)(1,1,2)[24] model for Lighting_Habitacion_Sensor

Figure 56: 1 hour forecast of Lighting_Habitacion_Sensor using ARIMA(1,0,0)(1,1,2)[24]

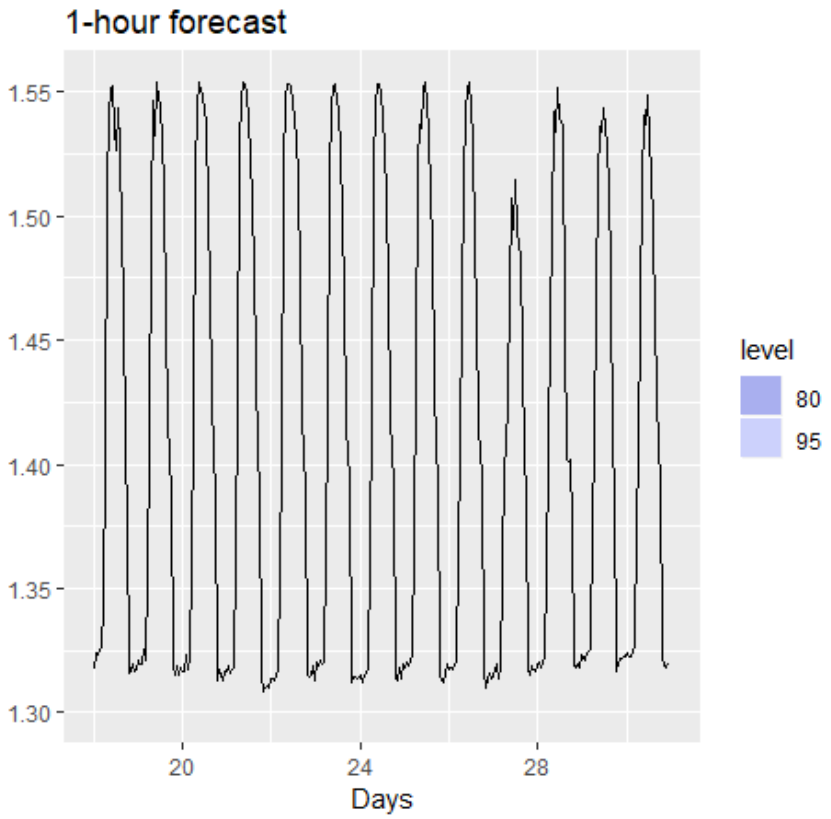


Figure 57: 24 hours forecast of Lighting_Habitacion_Sensor with ARIMA(1,0,0)(1,1,2)[24] with the 80% and 95%

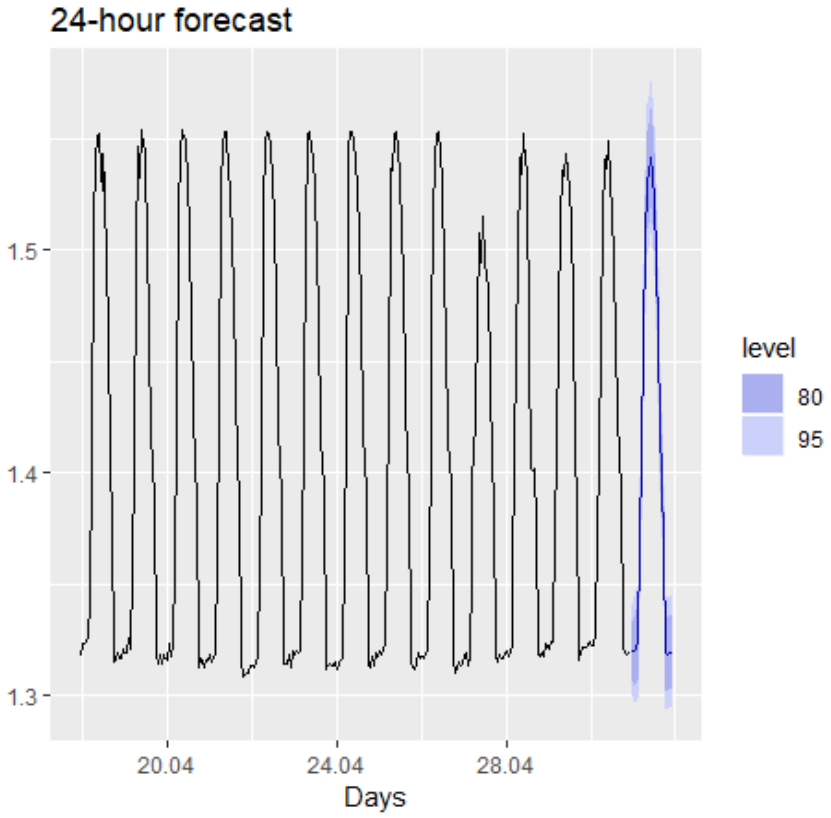


Figure 58: RMSEs score per variable with the selected methods

CrossValidation (tsCV)	Humedad Comedor Sensor	Humedad Exterior Sensor	Temperature Comedor Sensor	CO2 Comedor Sensor	Lighting Habitacion Sensor
RMSE1	0,6959974	2,242156	0,1300184	11,03737	0,01958917
RMSE24	3,99	5,637178	2,848945	12,4053	0,1249433
RMSE120	13,48346	11,48663	13,68994	12,5493	0,5791705