Code ▾

Research project

Abstract

This project investigated the efficiency of two time series forecasting models in predicting consumer price indices in the United States.

We build two models to make predictions.

1. A 2-stage SARIMAX-SARIMAX forecasting model.
2. Prophet model from Facebook.

To evaluate the models, machine learning techniques were used by separating the data into 2 datasets: training data and test data. RMSE was used as the evaluation measure.

In general, we can say that both models performed acceptably in their predictions, but the first SARIMAX-SARIMAX model gave better results than the second model, the Prophet model.

Project objectives

To compare the effectiveness of 2 models for predicting the consumer price index.

The **first model** is a two-stage prediction model of the consumer price index based on the quantity of each of the raw materials used to generate electricity.

- In the first stage we will try to forecast the price of electricity based on the historical data of the price itself and additionally incorporate information in the model about the raw materials used for generation.
- In the second stage we will use the results of the previous stage to build a second model that will lead us to predict the consumer price index using the historical data of this index and the price of electricity from the previous stage.

The **second model** is based on a method known as ***Facebook Prophet***, whose name comes from the fact that it was created by Facebook to make time series predictions.

Introduction

Electricity generation worldwide comes from different sources such as coal, oil, natural gas and various other sources. In many countries this diversification is quite remarkable, however we must mention that for many years the main source of generation in many countries has been coal. However, for some time now, the need to substitute coal as a raw material has been raised due to the environmental consequences that this entails.

The data were obtained from the U.S. Energy Information Administration website http://www.eia.gov/ (http://www.eia.gov/).

In fact, after some research we decided to work with data from the United States because in many other countries there was not enough historical data needed to work with what is intended in this project, there were even countries that could only obtain up to 24 data, which seemed to us very deficient to work with time series.

Now we are going to load the data.

We will work with the data to prepare it for processing, it should be noted that when reading the data the reading begins in row 205 which represents the month of January 1989, this is because from this date we have all the data of all the columns since before this date there are several columns where there is no historical information and it does not make sense to impute the data since we are talking about missing 200 consecutive data. As we are going to perform, among other things, a regression, there cannot be empty cells.

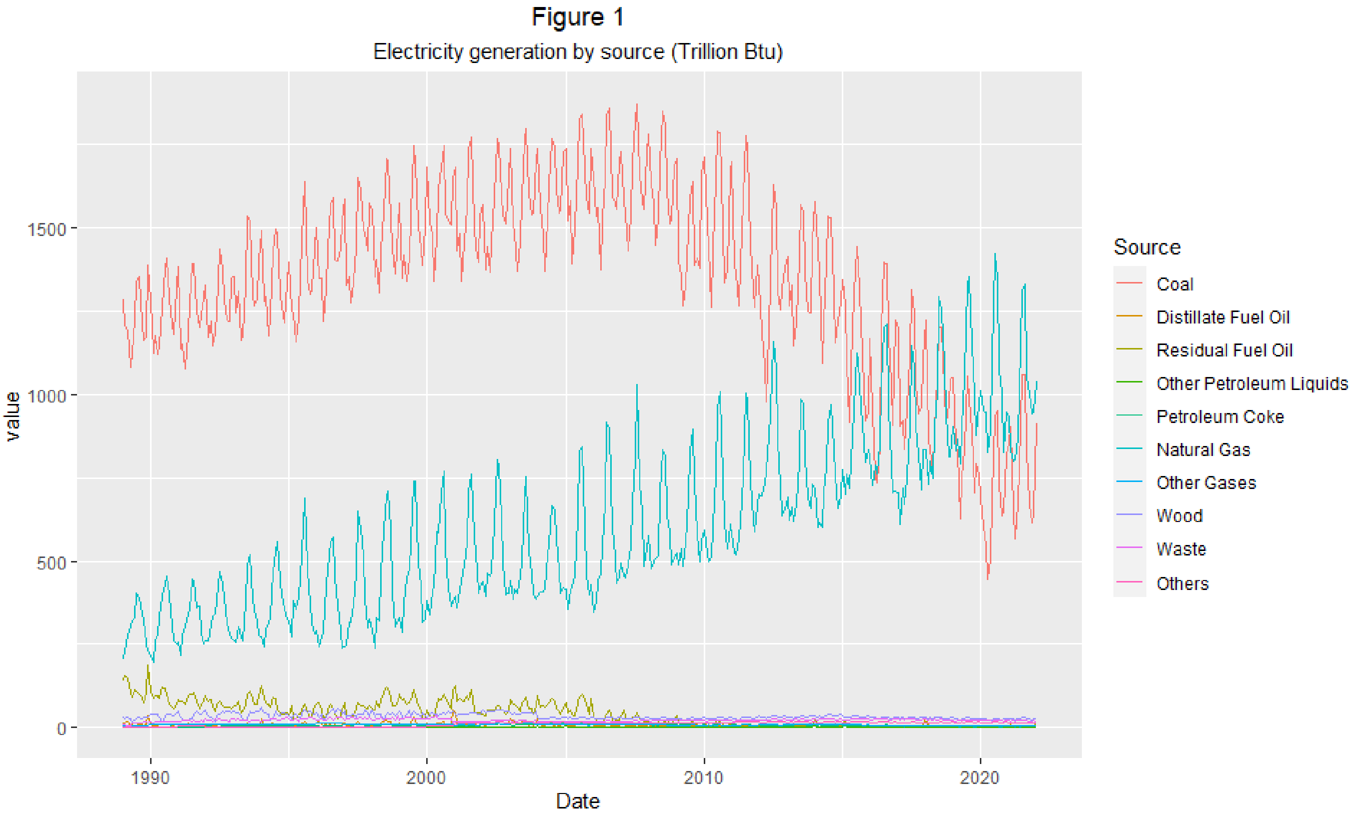
Although in a regression the variables can have different units without any problem to make a visual inspection and a fair comparison we will transform the different sources of electricity generation in BTU's. In this way we can know how many BTU's come from each source and thus we can make a more simple and direct comparison.

The units for all numerical values will then be trillion BTU's.

Hide

```
library("openxlsx")
dfSources <- read.xlsx(xlsxFile = "Consumption_of_Combustible_Fuels_for_Electricity_Generatio
n.xlsx", sheet = 1, colNames = FALSE,
                       rowNames = FALSE, startRow = 205)

dfSources[,1] = as.Date(dfSources[,1], origin = "1899-12-30")

colnames(dfSources ) <- c('Date', 'Coal','Distillate Fuel Oil','Residual Fuel Oil', 'Other Pe
troleum Liquids', 'Petroleum Coke', 'Total_Petroleum', 'Natural Gas', 'Other Gases', 'Wood',
'Waste', 'Others')

dfSources <- subset (dfSources[-c(7)])
# Thousand Short Tons to Trillion Btu Conversion
dfSources[,2]= dfSources[,2]*18856000/1000000000
dfSources[,6]= dfSources[,6]*18856000/1000000000
# Thousand Barrels to Trillion Btu Conversion
dfSources[,3]= dfSources[,3]*5691000/1000000000
dfSources[,4]= dfSources[,4]*5691000/1000000000
dfSources[,5]= dfSources[,5]*5691000/1000000000
# Billion Cubic Feet to Trillion Btu Conversion
dfSources[,7]= dfSources[,7]*1037/1000
```

Let us first look at the series representing the consumption of the different sources to generate electricity.



As we can see, natural gas has been gradually replacing coal as the main source of electricity generation, which was something we expected to see and which is also consistent with what is said about stopping the use of coal due to the environmental consequences of using this raw material.

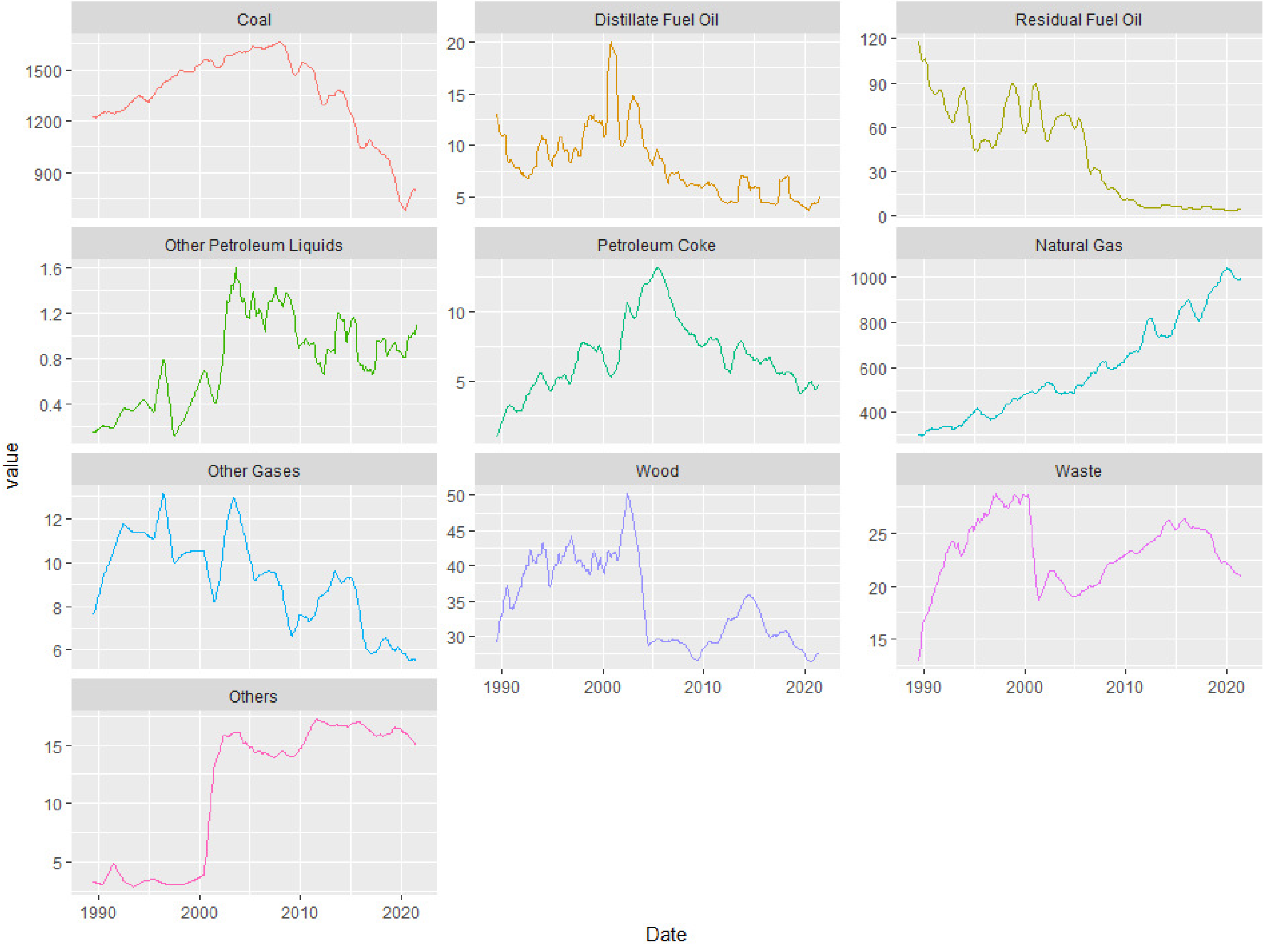
Let’s go a little deeper into this exploratory analysis, considering that it would be interesting to observe the trends of the different sources.

Hide

```
library("ggplot2")
library("reshape2")
dfTrends = as.data.frame(dfSources[ , c("Date")])
colnames(dfTrends)[1] = "Date"
for (i in seq(2, ncol(dfSources), 1)) {
tsdata <- ts(dfSources[,i], start = c(1989, 1), end = c(2022, 1), frequency = 12)
ddata <- decompose(tsdata, "additive")
dfTrends[,i] <- ddata$trend
colnames(dfTrends)[i] = colnames(dfSources)[i]
}
CombinedTrends <- melt(dfTrends, id.vars = "Date")
colnames(CombinedTrends )[2] = "Source"
ggplot(CombinedTrends,
       aes(x = Date,
           y = value, col = Source)) +
           geom_line() +
           facet_wrap(~Source, ncol = 3, scales = "free_y") +
           theme(legend.position = "none") +
           ggtitle("Figure 2",
           subtitle = "Electricity generation Trends by source (Trillion Btu)") +
           theme(plot.title = element_text(hjust = 0.5),
           plot.subtitle = element_text(hjust = 0.5))
```

Figure 2

Electricity generation Trends by source (Trillion Btu)



Hide

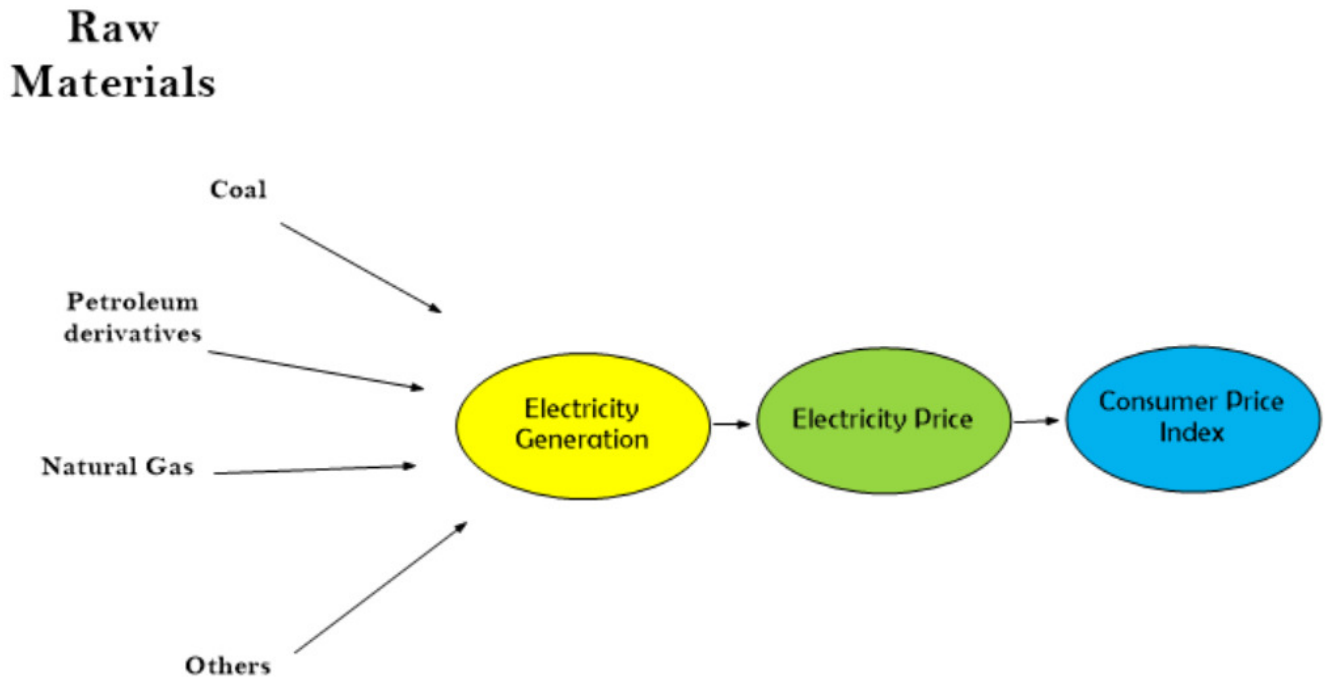
```
rm("dfTrends","CombinedTrends","ddata","tsdata", "i")
```

In these graphs we can see more clearly the trends of use for the different sources in electricity generation, however we must clarify that the 2 main sources are natural gas and coal and although as mentioned before natural gas has been gradually replacing coal as the first source of electricity generation, coal still continues to occupy second place and by far as could be seen in Figure 1(Generation from other sources remains insignificant compared to the two main sources).

In this first stage of our procedure consists of establishing an electricity price prediction model based on the historical electricity prices themselves, which would originally be an ARIMA model, but in this case we will additionally incorporate external variables to try to better explain the behavior of these prices. An ARIMA model to which external variables are incorporated is known as ARIMAX[1], however if the series has seasonality then it would be a SARIMAX[1] model.

> [1]Arunraj, Nari & Ahrens, Diane & Fernandes, Michael. (2016). Application of SARIMAX Model to Forecast Daily Sales in Food Retail Industry. International Journal of Operations Research and Information Systems. 7. 1-21. 10.4018/IJORIS.2016040101.

The general idea behind this first model is to analyze how the decision regarding the use of raw materials to generate electricity affects the price of electricity, which in turn affects industries and has repercussions on the final prices paid by consumers, affecting their daily lives.



Now let’s load the electricity price and CPI data and attach them as additional columns to our main data frame.

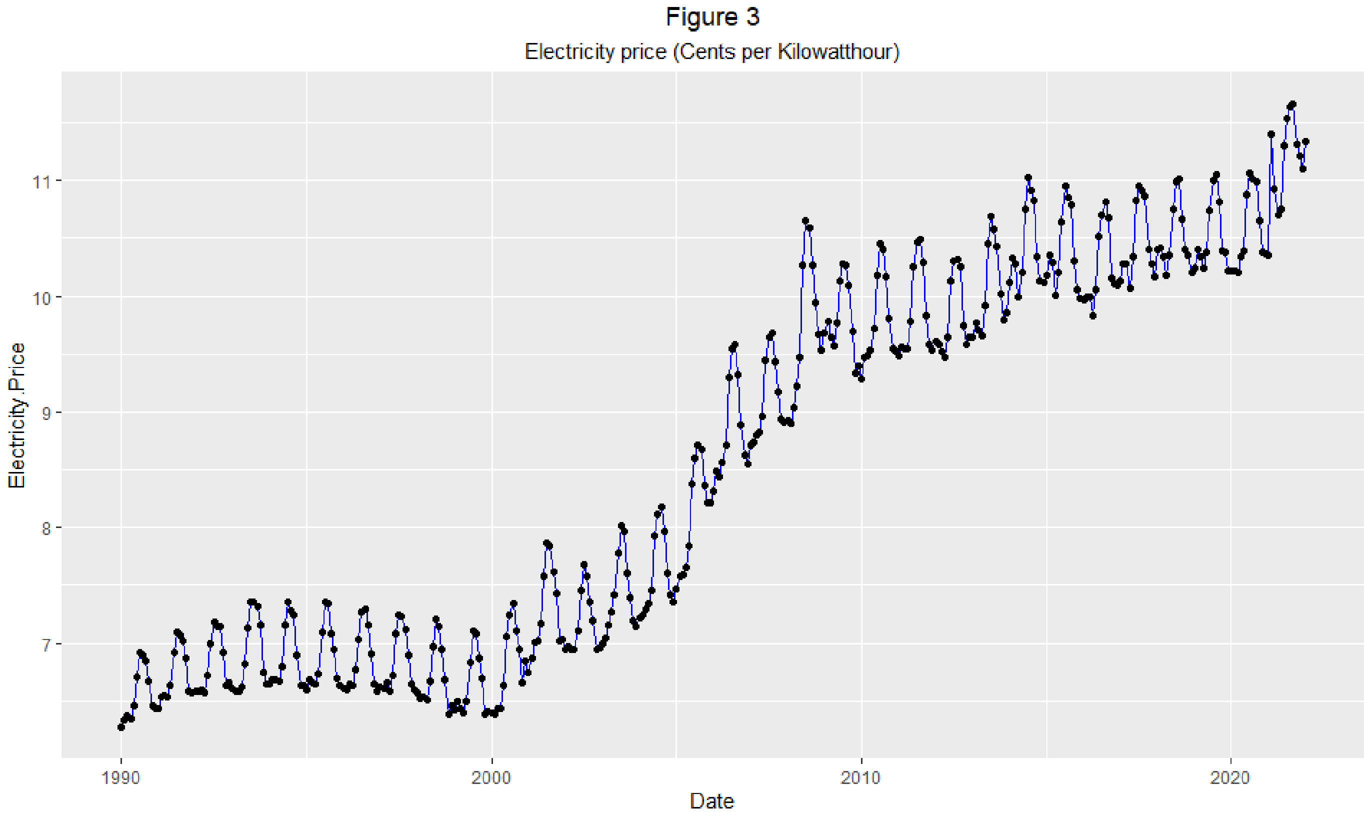
Hide

```
library("openxlsx")
library("reshape2")
Pricedf <- read.xlsx(xlsxFile = "Average_Retail_Prices_of_Electricity.xlsx", sheet = 1, colNames = FALSE, rowNames = FALSE, startRow = 181)
Pricedf <- as.data.frame(Pricedf[ , c(7)])
colnames(Pricedf) = "Electricity.Price"
Tempdf <- read.xlsx(xlsxFile = "ConsumerPriceIndex.xlsx", sheet = 1, colNames = TRUE, rowNames = FALSE, startRow = 12)
Tempdf <- Tempdf[ , c(-14,-15)]
Tempdf = as.data.frame(t(Tempdf))
Tempdf <- Tempdf[ -c(1) , ]
Tempdf <- Tempdf[  , -c(1:77)]
Tempdf = melt(Tempdf)
Tempdf <- Tempdf[ -c(386:nrow(Tempdf)) , ]
dfSources = dfSources[dfSources$Date >= "1990-01-01", ]
dfSources <- cbind(dfSources, Pricedf, Tempdf[,2])
colnames(dfSources)[13] = "CPI"
rm("Pricedf","Tempdf")
```

Now let’s explore a little bit our series of electricity prices.

Hide

```
library("ggplot2")

ggplot(dfSources,
       aes(x = Date,
           y = Electricity.Price)) +
           geom_line(color="blue") +
           geom_point() +
           ggtitle("Figure 3",
           subtitle = "Electricity price (Cents per Kilowatthour)") +
           theme(plot.title = element_text(hjust = 0.5),
           plot.subtitle = element_text(hjust = 0.5))
```



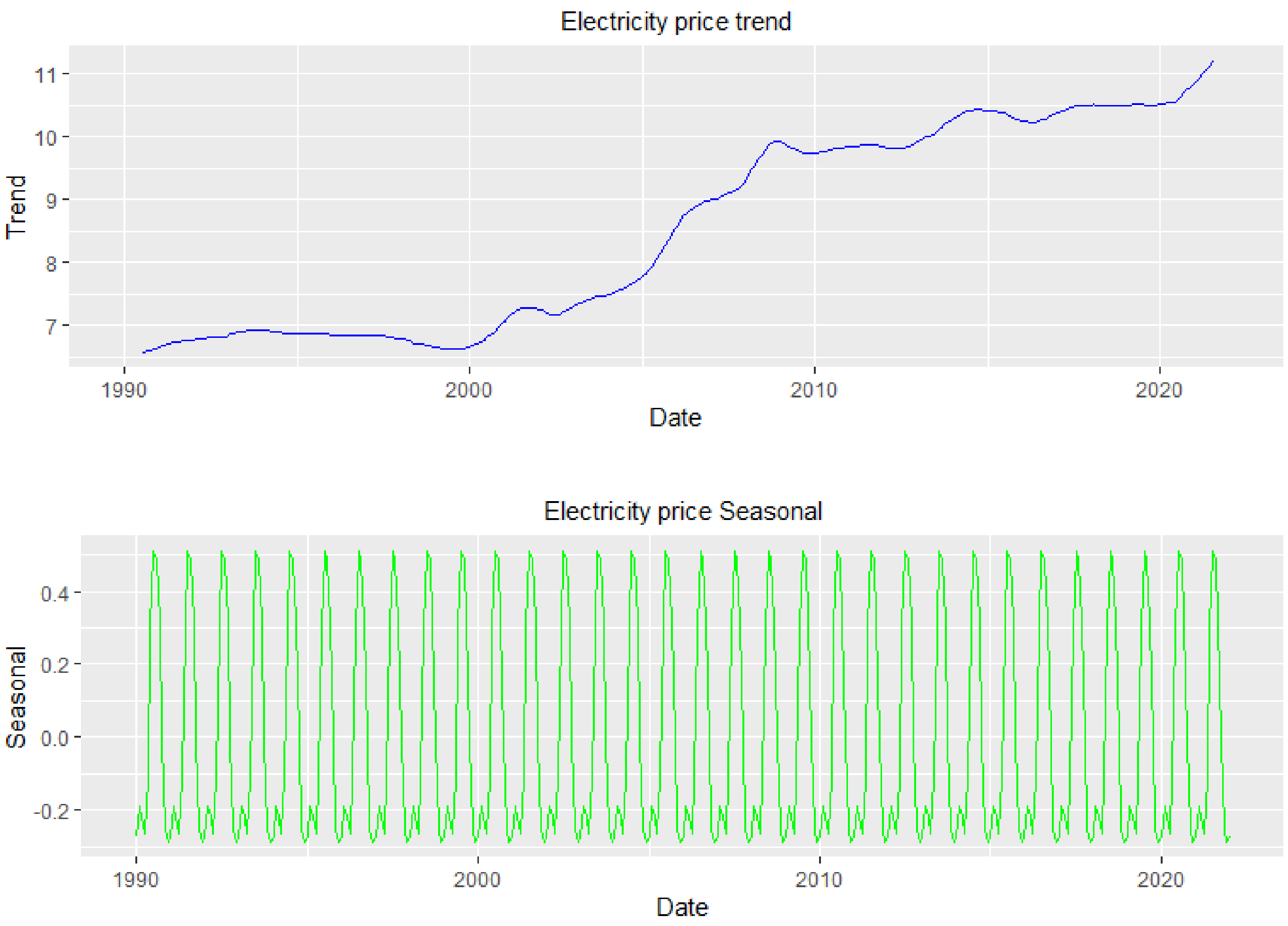
On scrutiny and at a glance, we can say that this time series has trend and seasonality.

Next we are going to decompose the series and show these components.

Hide

```
library("gridExtra")
library("ggplot2")
dfElectricity = ts(dfSources[,"Electricity.Price"], start = c(1990, 1), end = c(2022, 1), frequency = 12)
DecomPrices = decompose(dfElectricity, type = "additive")
Tempdf = as.data.frame(dfSources$Date)
Tempdf = cbind(Tempdf,DecomPrices$trend,DecomPrices$seasonal)
colnames(Tempdf) <- c("Date","Trend","Seasonal" )

plot1 <- ggplot(Tempdf,
       aes(x = Date,
           y = Trend)) +
           geom_line(color="blue") +
           ggtitle("Figure 4",
           subtitle = "Electricity price trend") +
           theme(plot.title = element_text(hjust = 0.5),
           plot.subtitle = element_text(hjust = 0.5))
plot2 <- ggplot(Tempdf,
       aes(x = Date,
           y = Seasonal)) +
           geom_line(color="green") +
           ggtitle(" ",
           subtitle = "Electricity price Seasonal") +
           theme(plot.title = element_text(hjust = 0.5),
           plot.subtitle = element_text(hjust = 0.5))

grid.arrange(plot1, plot2, ncol=1, nrow = 2)
```

Figure 4



Hide

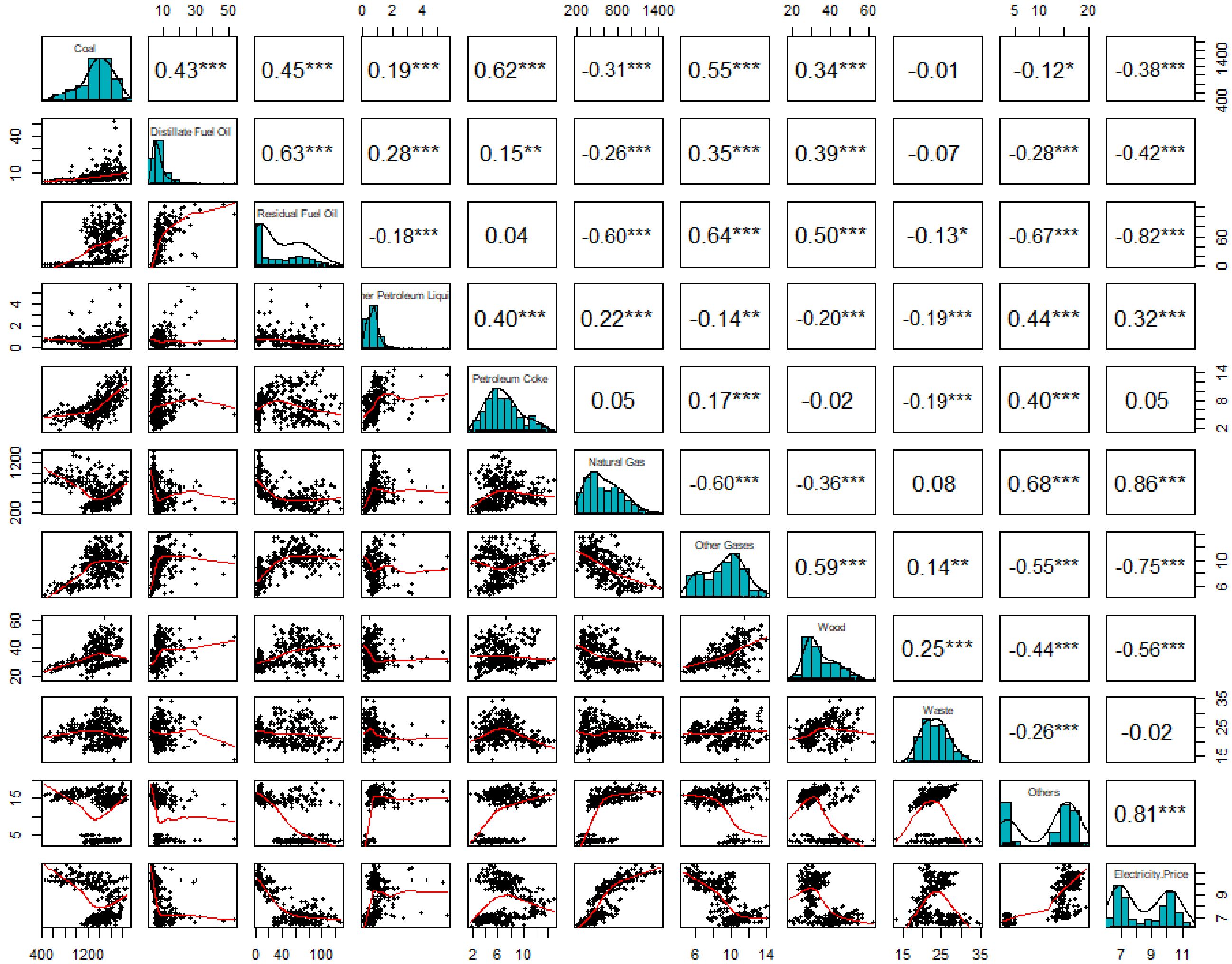
```
rm("DecomPrices","plot1", "plot2", "Tempdf", "dfElectricity")
```

To make predictions about the price of electricity we will then resort to a SARIMAX model but remembering that the final decision of the model will be based on the results of the auto.arima() function.

As our intention is to add external variables to the prediction model, we will create a correlation matrix where we will show the correlations between the variables and electricity price, to see if these correlations are significant and therefore worth adding to the model.

Hide

```
library(psych)
pairs.panels(dfSources[,2:12],
             method = "pearson",
             hist.col = "#00AFBB",
             density = TRUE,
             ellipses = FALSE,
             stars = TRUE
)
```



The last column represents the electricity price, in this column we can check if there is a significant relationship with the variables that we want to incorporate as external variables to the arima model. The numbers represent the correlations and if they are accompanied by asterisks it implies that the relationship is considered significant. As we can see, there is a significant correlation with almost all the variables we wish to incorporate.

However, to obtain the best model we will rely on the auto.arima() function.

Hide

```
library(forecast)
options(width = 120)

dfElectricity = ts(dfSources[,"Electricity.Price"], start = c(1990, 1), end = c(2015, 8), fre
quency = 12)

VarReg <- dfSources[,2:11]
EndData = nrow(VarReg)*0.8
VarRegTrain <- VarReg[1:EndData,]
EndData = EndData + 1
VarRegTest <- VarReg[EndData:nrow(VarReg),]
VarRegTrain <- as.matrix.data.frame(VarRegTrain)
VarRegTest <- as.matrix.data.frame(VarRegTest)

PriceSarimax <- auto.arima(dfElectricity, xreg = VarRegTrain, seasonal = TRUE, stepwise = FAL
SE,  approximation = FALSE)

summary(PriceSarimax)
```

```
Series: dfElectricity
Regression with ARIMA(1,1,0)(1,0,0)[12] errors

Coefficients:
         ar1    sar1   Coal  Distillate Fuel Oil  Residual Fuel Oil  Other Petroleum Liquids
Petroleum Coke
      0.1620  0.8721  0e+00               0.0009             3e-04                   0.0016
0.0063
s.e.  0.0605  0.0286  1e-04               0.0011             4e-04                   0.0070
0.0036
      Natural Gas  Other Gases    Wood    Waste   Others
            2e-04       0.0008  -1e-04  -0.0004  -0.0106
s.e.        1e-04       0.0066   8e-04   0.0016   0.0056

sigma^2 estimated as 0.007859:  log likelihood=305.79
AIC=-585.59   AICc=-584.34   BIC=-537.14

Training set error measures:
                      ME      RMSE        MAE       MPE      MAPE      MASE         ACF1
Training set 0.001180985 0.0867606 0.06560126 0.0147199 0.8064042 0.2992906 -0.006198518
```

Hide

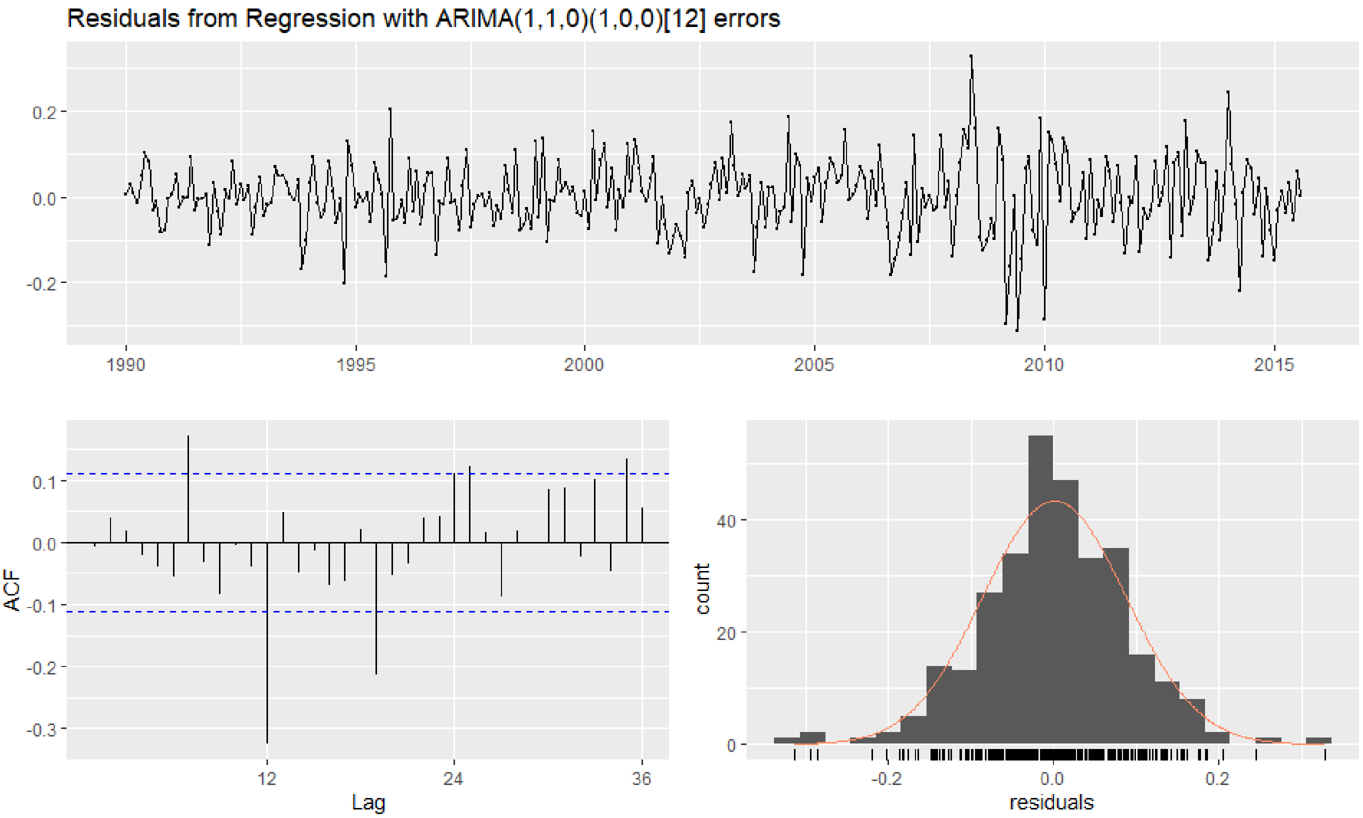
```
rm("VarReg","EndData")
```

Now we see a little bit of the behavior of the errors in our model. Recall that for the approximation to be good, the errors must have the behavior of a normal variable with mean 0.

Hide

```
checkresiduals(PriceSarimax)
```

```


    Ljung-Box test

data:  Residuals from Regression with ARIMA(1,1,0)(1,0,0)[12] errors
Q* = 74.899, df = 12, p-value = 3.84e-11

Model df: 12.   Total lags used: 24
```

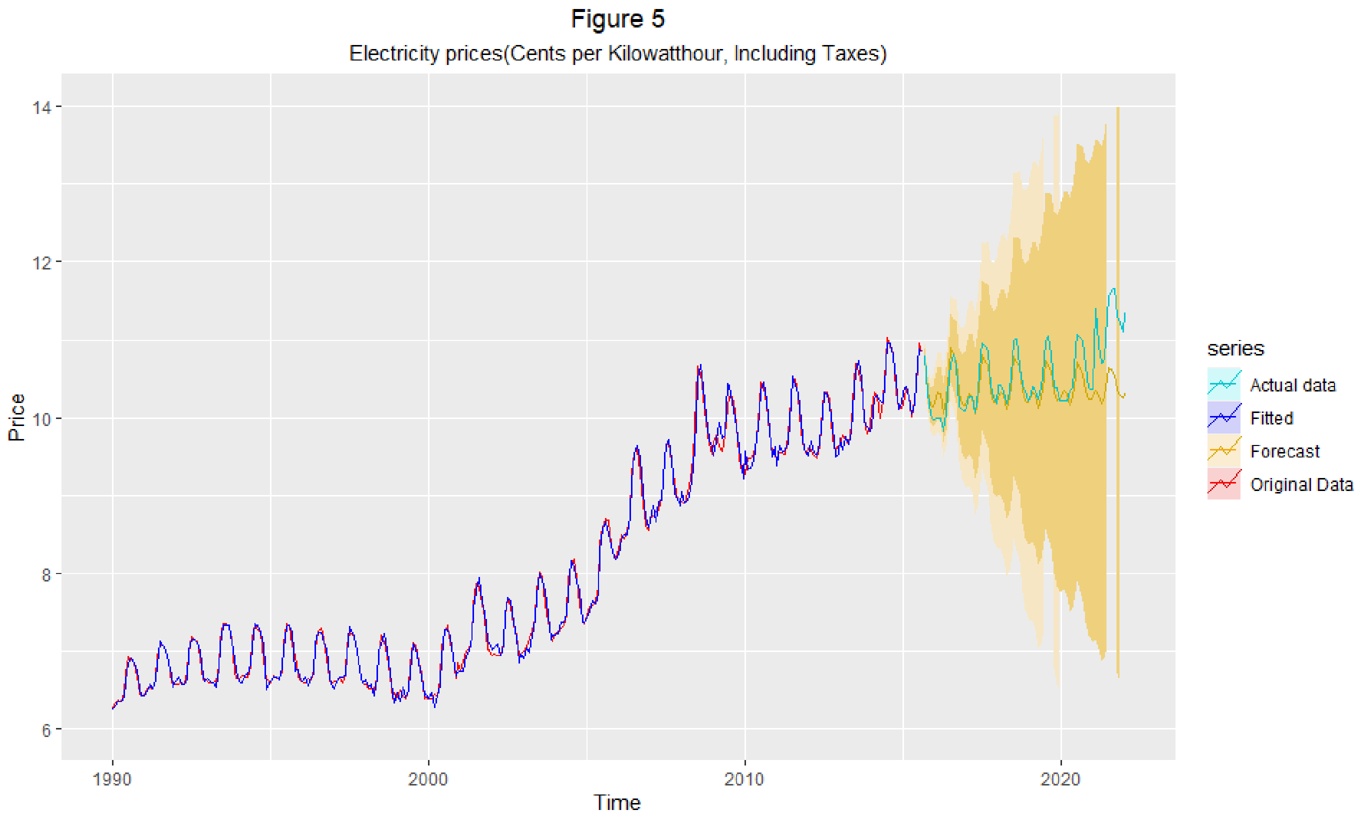


Analyzing the obtained results we can say that the errors have an approximately normal behavior with mean 0. Of course our model is not perfect but let's say that it is an acceptable approximation. As we can see in the ACF graph and also in the Ljung-Box test results there is autocorrelation in the errors however we can still make predictions with our model only that the prediction intervals are more imprecise although we will inspect this with our test data.

Now we are going to see graphically how the series created with our model(blue) behaves with respect to the original values (red) and additionally we will add the predictions (orange).

Hide

```
library("ggplot2")
library(forecast)

dfElectricity2 = ts(dfSources[309:nrow(dfSources),"Electricity.Price"], start = c(2015, 9), e
nd = c(2022, 1), frequency = 12)

forecastSet<-forecast(PriceSarimax, xreg = VarRegTest)

autoplot(dfElectricity, series="Original Data") +
          ggtitle("Figure 5",
                  subtitle = "Electricity prices(Cents per Kilowatthour, Including Taxes)")
+
          theme(plot.title = element_text(hjust = 0.5),
          plot.subtitle = element_text(hjust = 0.5)) +
          autolayer(forecastSet, series="Forecast") +
          autolayer(fitted(forecastSet), series="Fitted") +
          autolayer(dfElectricity2, series="Actual data") +
          ylab("Price") +
          ylim(6,14) +
       scale_color_manual(values=c("turquoise3", "blue", "orange","red"))
```



Hide

```
VarRegTest2 = forecastSet$mean

rm("forecastSet","PriceSarimax","VarRegTest","VarRegTrain",
   "dfElectricity","dfElectricity2")
```

First, as for our model we can say that it has a quite acceptable behavior, we can see how the values in blue (obtained by our model) are almost superimposed with the values in red (original data) and also in the period where the predictions were made were quite acceptable, always remembering that as you move away in time these predictions tend to be more inaccurate(Recall that predictions were made for 77 periods or 77 months).

Bearing in mind that our procedure is a two-stage one, we now move on to the second stage.

In this second stage we are now going to build a model to be able to analyze the **consumer price index** where in addition to using its historical data we are going to use as an external variable the electricity prices, i.e. using the information resulting from the first model.

Consequently, we are now going to build a new ARIMAX(SARIMAX) model.

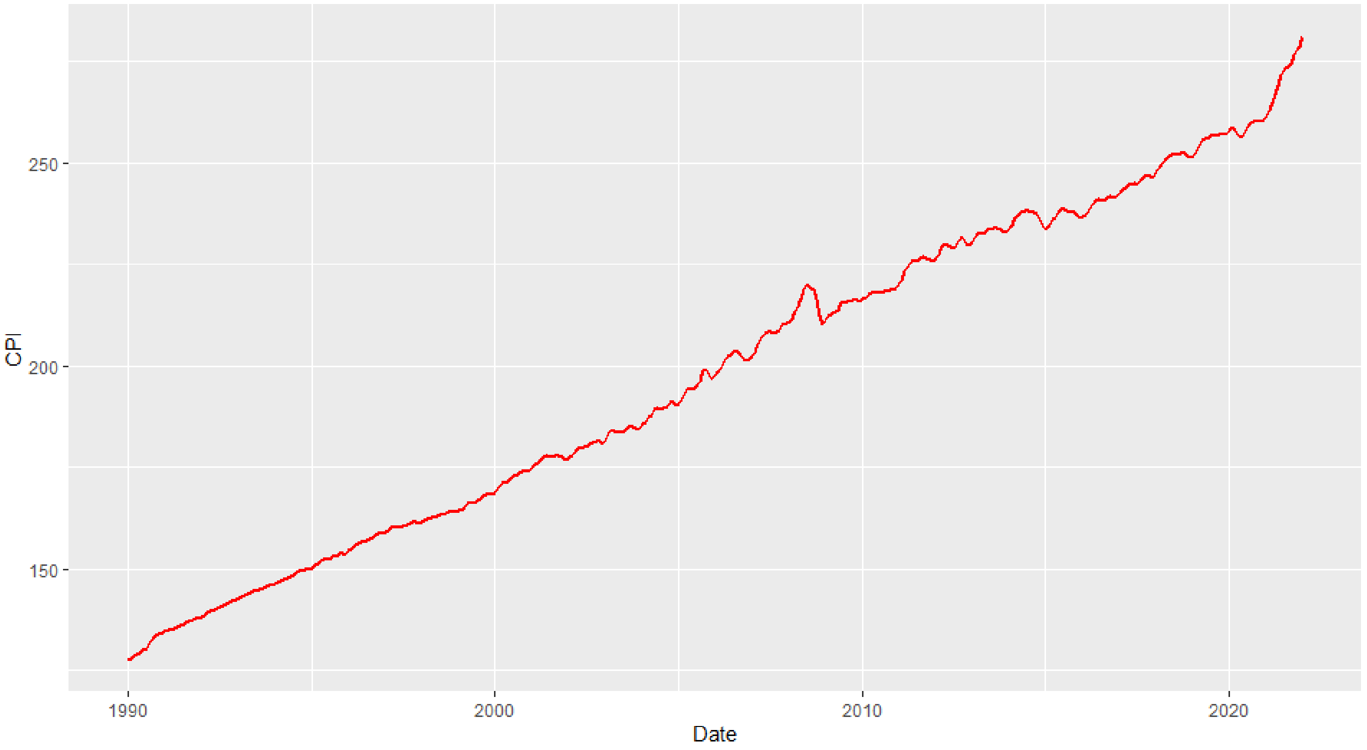
As we did with the electricity price series, we will first inspect the consumer price index series.

Hide

```
library("ggplot2")

ggplot(dfSources,
       aes(x = Date,
           y = CPI)) +
           geom_line(color="red", size = 1) +
           ggtitle("Figure 6",
           subtitle = "Consumer Price Index") +
           theme(plot.title = element_text(hjust = 0.5),
           plot.subtitle = element_text(hjust = 0.5))
```

Figure 6

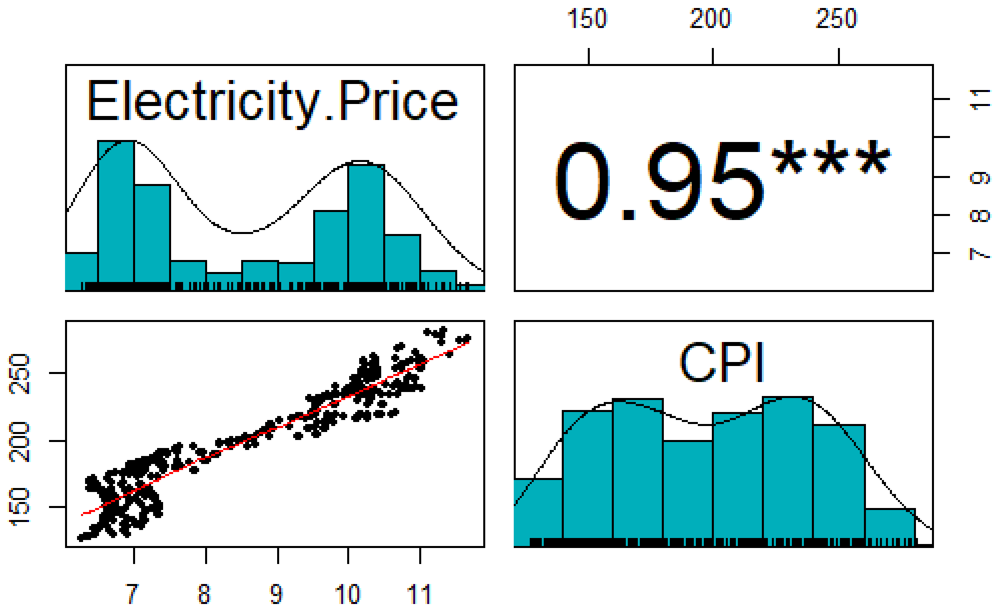
Consumer Price Index



Although there is no significant evidence in the graph of a seasonal component we will let auto.arima() do its job to find the best model. But first, as at this stage the model will be used to predict the consumer price index using as an external variable the electricity prices of the previous model, we will do the same as we did before to find out if it is reasonable to incorporate this external variable in the ARIMA model, let us verify if any type of relationship between these two variables is detectable.

Hide

```
library(psych)
pairs.panels(dfSources[,12:13],
             method = "pearson",
             hist.col = "#00AFBB",
             density = TRUE,
             ellipses = FALSE,
             stars = TRUE
)
```



As is quite noticeable, there is a high relationship between these two variables.

Now let’s proceed to find our model.

Hide

```
library(forecast)
options(width = 120)

dfIndex = ts(dfSources[,"CPI"], start = c(1990, 1), end = c(2015, 8), frequency = 12)

dfIndex2 = ts(dfSources[309:nrow(dfSources),"CPI"], start = c(2015, 9), end = c(2022, 1), frequency = 12)

VarReg <- as.data.frame(dfSources[,"Electricity.Price"])
EndData = nrow(VarReg)*0.8
VarRegTrain <- as.data.frame(VarReg[1:EndData,])
colnames(VarRegTrain) = c("Electricity.Price")
EndData = EndData + 1
VarRegTest2 <- data.matrix(VarRegTest2)
colnames(VarRegTest2) = c("Electricity.Price")

VarRegTest <- as.data.frame(VarReg[EndData:nrow(VarReg),])
colnames(VarRegTest) = c("Electricity.Price")

VarRegTrain <- as.matrix.data.frame(VarRegTrain)
VarRegTest <- as.matrix.data.frame(VarRegTest)

CPImodel <- auto.arima(dfIndex, xreg = VarRegTrain,
                       stepwise = FALSE,
                       approximation = FALSE)

CPImodel
```

```
Series: dfIndex
Regression with ARIMA(1,1,2)(0,1,2)[12] errors

Coefficients:
          ar1     ma1     ma2     sma1    sma2  Electricity.Price
      -0.5166  1.0428  0.3763  -0.9885  0.1324             0.4198
s.e.   0.1996  0.1931  0.0993   0.0639  0.0628             0.3217

sigma^2 estimated as 0.2638:  log likelihood=-227.6
AIC=469.21   AICc=469.6   BIC=495.02
```

Hide

```
accuracy(forecast(CPImodel, xreg = VarRegTest2) ,x = dfIndex2)
```

```
                     ME      RMSE       MAE         MPE      MAPE       MASE       ACF1 Theil
's U
Training set -0.0218249 0.4974899 0.3345684 -0.01294253 0.1732412 0.07537521 -0.0055514        
NA
Test set      3.2378868 5.8131963 3.7309762  1.20795383 1.4133917 0.84055491  0.8951578  5.74
4101
```

Hide

```
rm("VarReg","EndData")
```

It is important to note the RMSE value obtained for the model using the test data.

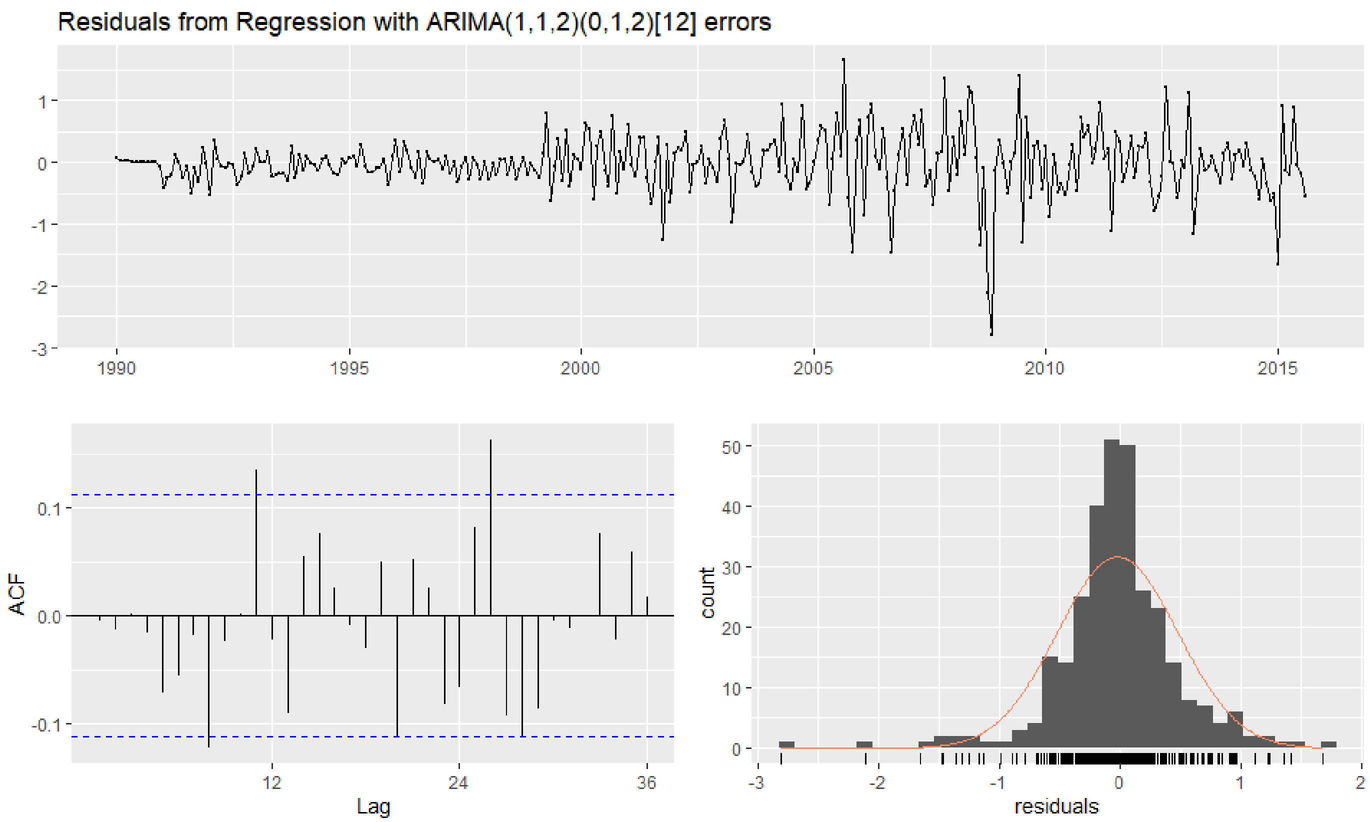
RMSE = 5.8113

Let us now review the behavior of the model’s errors

Hide

```
checkresiduals(CPImodel)
```

```


    Ljung-Box test

data:  Residuals from Regression with ARIMA(1,1,2)(0,1,2)[12] errors
Q* = 29.768, df = 18, p-value = 0.03977

Model df: 6.   Total lags used: 24
```



Again, an acceptable approximation, however, it is necessary to take into account what was commented in the previous series on the autocorrelation of the errors

Hide

```
library("ggplot2")
library(forecast)

forecastSet<-forecast(CPImodel, xreg = VarRegTest2)

autoplot(dfIndex, series="Original Data") +
           ggtitle("Figure 7",
                   subtitle = "Consumer Price Index(Base Period:1982-84=100)") +
           theme(plot.title = element_text(hjust = 0.5),
           plot.subtitle = element_text(hjust = 0.5)) +
           autolayer(forecastSet, series="Forecast") +
           autolayer(fitted(forecastSet), series="Fitted") +
           autolayer(dfIndex2, series="Actual data") +
           ylab("Consumer Price Index") +
        scale_color_manual(values=c("turquoise3", "blue", "orange","red"))
```

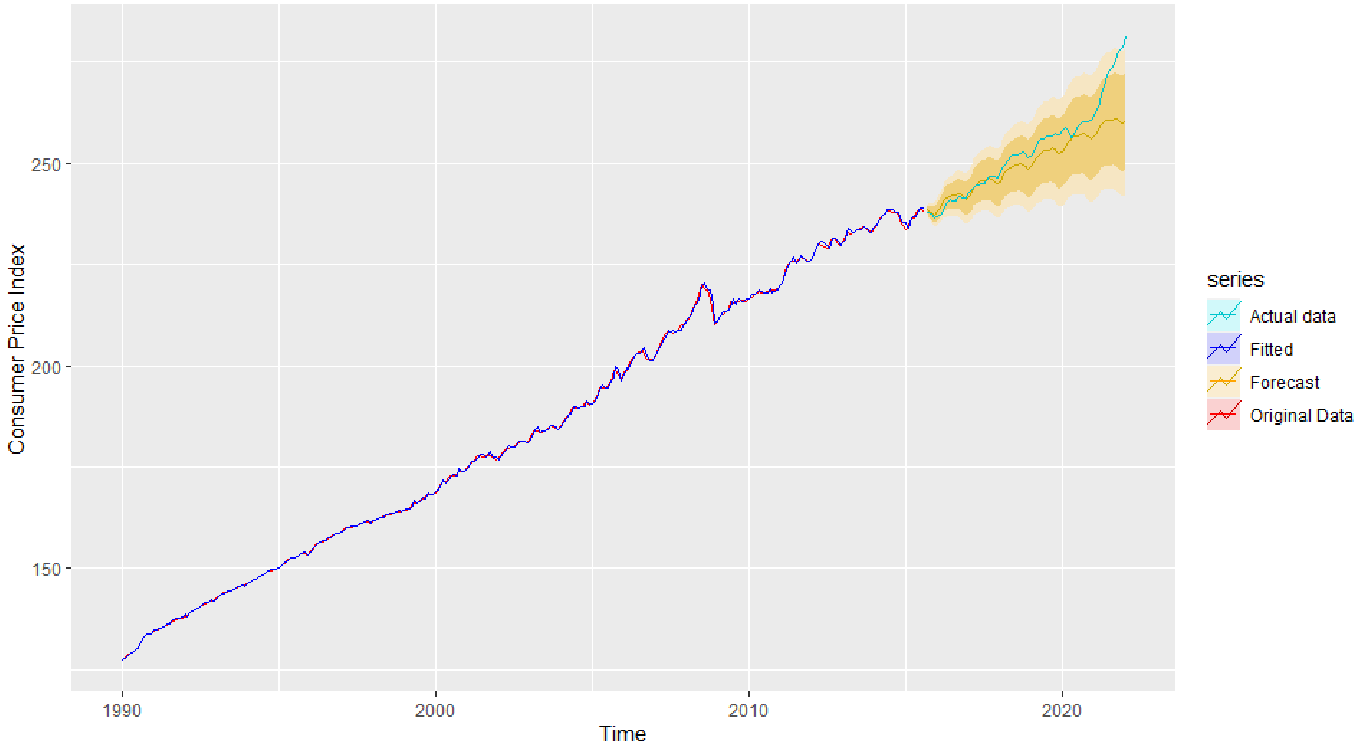


Figure 7

Consumer Price Index(Base Period:1982-84=100)

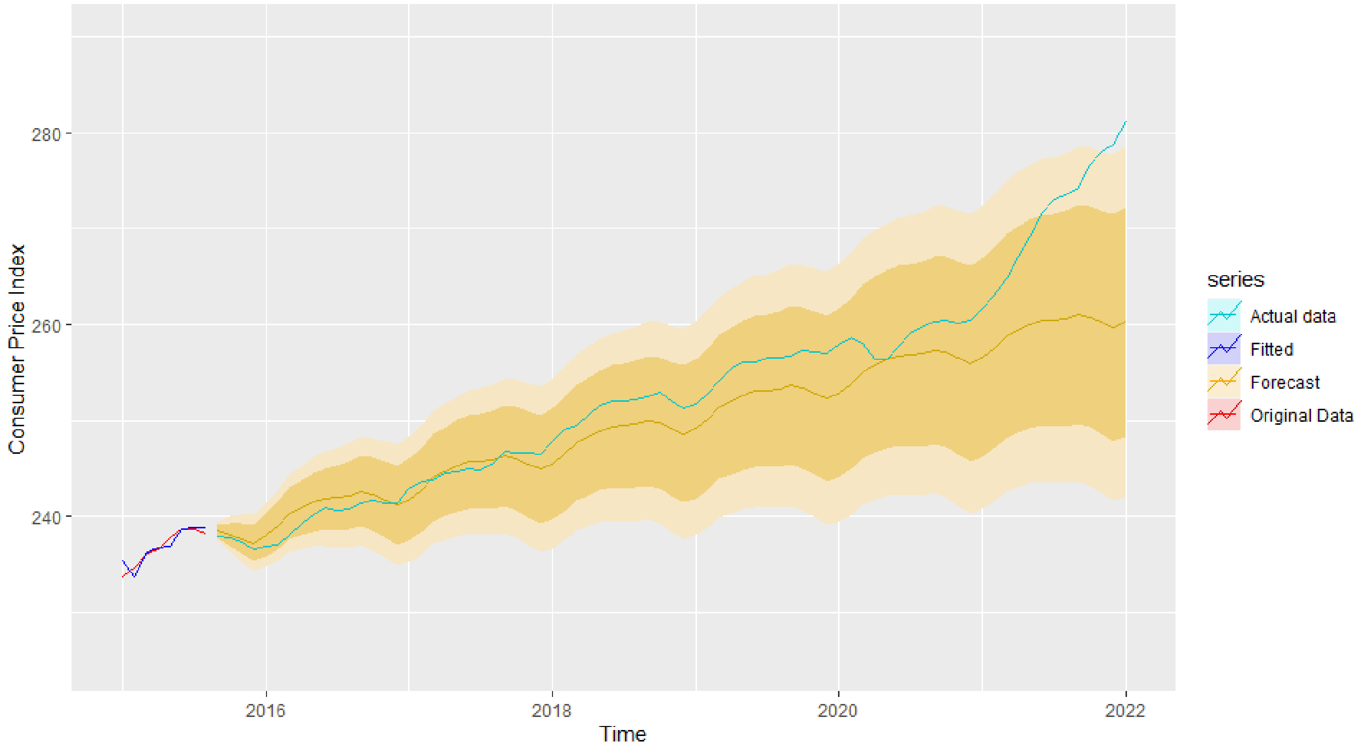
Let’s take a closer look at the predicted values for the test data.

Hide

```
library("ggplot2")
library(forecast)

autoplot(dfIndex, series="Original Data") +
          ggtitle("Figure 8",
                 subtitle = "Consumer Price Index(Base Period:1982-84=100)") +
          theme(plot.title = element_text(hjust = 0.5),
          plot.subtitle = element_text(hjust = 0.5)) +
          autolayer(forecastSet, series="Forecast") +
          autolayer(fitted(forecastSet), series="Fitted") +
          autolayer(dfIndex2, series="Actual data") +
          ylab("Consumer Price Index") +
          xlim(2015,2022) +
          ylim(225,290) +
          scale_color_manual(values=c("turquoise3", "blue", "orange","red"))
```

Figure 8

Consumer Price Index(Base Period:1982-84=100)



Hide

```
rm("forecastSet","CPImodel","VarRegTest","VarRegTrain"," VarRegTest2","dfIndex","dfIndex2")
```

As we can see let’s say that until January 2018 the predictions were acceptable, considering that the predictions started in month 10 of the year 2015 we could say that for more than 2 years our model predicted within an acceptable range.

The intention now is to be able to compare this result with the result obtained using a relatively recent method created by Facebook known as “Facebook Prophet”[2][3].

> [2]Foxworthy, John (2020, Jul 3). The Facebook Prophet Prediction Model and Product Analytics. ScienceBlog. https://foxworthy-8036.medium.com/the-facebook-prophet-prediction-model-and-product-analytics-a1db05fbe454 (https://foxworthy-8036.medium.com/the-facebook-prophet-prediction-model-and-product-analytics-a1db05fbe454)
>
> [3]Time Series Analysis using Facebook Prophet in R Programming. (2020, Jul 22). In GeeksforGeeks. https://www.geeksforgeeks.org/time-series-analysis-using-facebook-prophet-in-r-programming/ (https://www.geeksforgeeks.org/time-series-analysis-using-facebook-prophet-in-r-programming/)

In practice, Facebook prophet has proven to be more efficient with series whose fluctuations are not exaggeratedly pronounced, as is the case here, but we must keep in mind that this method is constantly being improved by Facebook.

The modeling of this method is quite simple as well as powerful.

Mathematically our model can be written in the following way:

y(t) = g(t) + s(t) + h(t) + e(t)

Where:

- g(t) **trend** models non-periodic changes
- s(t) **seasonality** represents periodic changes
- h(t) **holidays** potentially irregular schedules
- e(t) **error** Random component

Prophet uses two possible models to estimate the trend:

- Logistic growth model

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

Where:

C → is the carry capacity

k → is the growth rate

m → is an offset parameter

- Piece-wise linear model

$$y = \begin{cases} \beta_0 + \beta_1\, x & x \leq c \\ \beta_0 - \beta_2\, c + (\beta_1 + \beta_2)\, x & x > c \end{cases}$$

Where:

c → is the trend change point

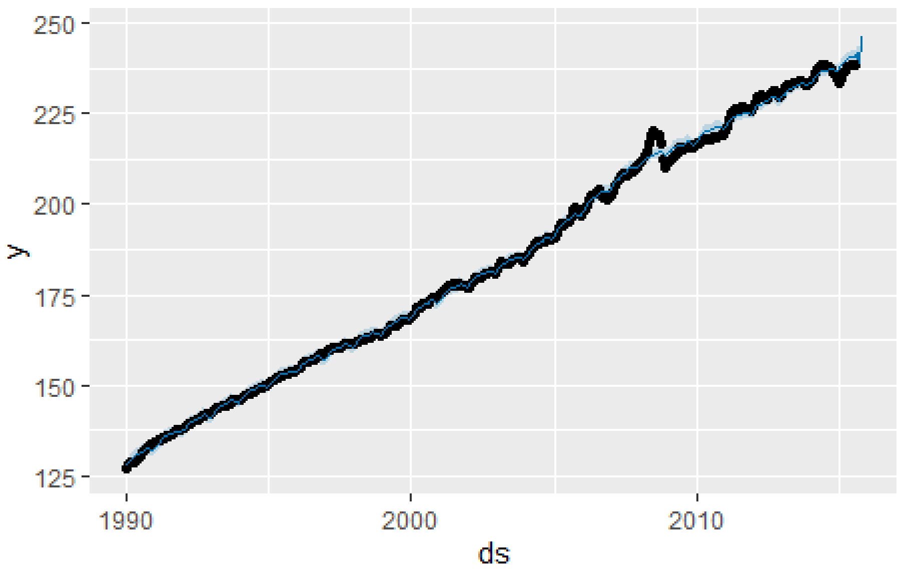
This process is fully automated using the ***Prophet library***.

To use this model, the data must be prepared in a specific way to be read by the function.

We will now prepare the data.

Hide

```
library(prophet)

Prophetdf = dfSources["Date"]
Prophetdf["y"] = dfSources["CPI"]

colnames(Prophetdf) <- c('ds', 'y')

EndData = nrow(Prophetdf)*0.8

ProphetdfTrain = Prophetdf[1:EndData,]
EndData = EndData + 1
ProphetdfTest <- Prophetdf[EndData:nrow(Prophetdf),]

modelProphet <- prophet(ProphetdfTrain)

futureDates <- make_future_dataframe(modelProphet, periods=77)

forecastSet <- predict(modelProphet, futureDates)

plot(modelProphet, forecastSet)
```



The blue line represents the values estimated by our model and the black line represents the actual values.

We can see that there are some small jumps between 2007, 2008 and 2009 approximately but the blue line does not follow these jumps and the prediction gets worse.

We will now calculate the RMSE of this Prophet model and compare it with the RMSE of the SARIMAX model obtained earlier, in both cases using the prediction period, i.e. the test dataset. I am not interested in the behavior during the training period, but who got better predictions.

Hide

```
RMSE = (sum((dfSources[309:nrow(dfSources),"CPI"]-forecastSet[309:nrow(forecastSet),"yhat"])^
2)/nrow(dfSources))^0.5

RMSE
```

```
[1] 6.820233
```

Hide

```
rm("forecastSet","futureDates","modelProphet","Prophetdf","ProphetdfTrain","ProphetdfTest", "
EndData", "RMSE")
```

The SARIMAX bietapic prediction model is our best model in this particular case as it obtained an RMSE of 5.81.

Conclusion

We were able to observe that there is a close, i.e. significant, relationship between final electricity prices and the amount of different raw materials used in the production of that electricity. Figure 5 shows the good performance of the first model in predicting electricity prices. Figure 8 also shows an acceptable performance of the second model in the prediction of the consumer price index using the information predicted from the first SARIMAX model, so that the predictions were generally good.

Finally, based on the results, we can say that the SARIMAX model performs better, at least in this particular case, than the Prophet model.

References

Bisgaard, S. & Kulahci, M (2011). TIME SERIES ANALYSIS AND FORECASTING BY EXAMPLE . John Wiley & Sons, Inc. .

Coghlan, A (2018). A Little Book of R For Time Series (Release 0.2).

SHMUELI, G. & LICHTENDAHL, K (2016). PRACTICAL TIMESERIES FORECASTING WITH R(2nd ed.). Axelrod Schnall publishers.

Robson, Winston. “The Math of Prophet.” Medium, Future Vision, 1 Oct. 2020, medium.com/future-vision/the-math-of-prophet-46864fa9c55a.

file	**lines**	**words**	**bytes**	**chars**	**nonws**	**path**
<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>
ResearchReport.Rmd	281	2191	13684	13684	11032	.

1 row