Proceedings of the International Conference on Industrial Engineering and Operations Management
Toronto, Canada, October 23-25, 2019

Comparative Analysis in Sales Forecasting: Classical Methods and Neural Networks

Ghita Benboubker, Dr. Ilham Kissani, Dr. Asmaa Mourhir
School of Science and Engineering
Al Akhawayn University
Ifran, Morocco
G.Benboubker@aui.ma
I.Kissani@aui.ma
A.Mourhir@aui.ma

Abstract

Forecasting has proven to be a useful tool that can constitute a competitive edge in today's competitive market. Indeed, forecasting deals with uncertainty by predicting the future either based on human judgement (qualitative forecasting) or based on historical data (quantitative forecasting). This paper explores and compares different quantitative forecasting methods for the sales of four sizes of a product belonging to a well-known company. These stock keeping units (SKUs) pertain to a carbonated beverage with high daily sales such that everyday hundreds of boxes are sold. To that end, this analysis focuses on one specific geographical area which is Fez, Morocco. Both the stock keeping units and the location were chosen with no prior motivation. In a first instance, we clean the raw data set. Then, we delve into exploring different classical and neural network forecasting methods, and comparing their accuracies to determine the most precise one, using the statistical programming language R. The results revealed that the Neural Network model outperforms the other methods.

Keywords
Predictive Forecasting, ARIMA, ETS, Neural Networks, R Programming.

1. Introduction

In the current market, forecasting provides a competitive edge to companies that wish to reduce the costs tied to their inventory while making sure that they can meet the demand for their products. This optimization process is possible thanks to forecasting methods for the demand, which enable businesses to control their inventory levels.

Forecasting can be defined as the scientific process through which a future parameter can be estimated using past data. In other words, even in a fast-changing environment, forecasts capture patterns and relationships to model the way the chosen subject in our environment is changing. According to Hyndman and Athanasopoulos (2018), the predictability of an event is mainly contingent upon three factors: the quantity of past data available, our understanding of the predictors that affect the predicted variable and the impact of the forecast on the forecasted variable, if any.

To illustrate the third factor, we consider the example of stocks that is inspired from the currency rate exchange example provided in the book. If the forecasts for such variable are publicized, people will follow the trend provided by the forecasts. In other words, if the forecasts indicate an increase in the price of stocks for a given company, people will scurry to buy them, making the prices go up. The forecast is said to be self-fulfilling (Hyndman and Athanasopoulos 2018).

Choosing the right forecasting method can only be achieved through understanding the type of data at hand and what it conveys. One classification of forecasting methods differentiates between qualitative and quantitative methods, which can be complementary in order to increase the accuracy of the forecast. Qualitative methods, also referred to as judgmental forecasts, are often the only option when no past data is available. On the other hand, quantitative or statistical methods rely on past data and they are more accurate when we can assume that some of the past trends will extend in the future (Hyndman and Athanasopoulos 2018).

© IEOM Society International

Another classification distinguishes between classical methods and Machine Learning (ML) algorithms. A study conducted by Makridakis et al. (2018) argues that complex and sophisticated methods do not provide more accuracy than classical methods for short-term times series forecasting. The study compares 10 Machine Learning algorithms and 8 classical forecasting methods on a total of 1045 univariate monthly data sets. The main finding of the study is that classical statistical forecasting methods are more accurate than ML methods. However, and as the authors admit to it, this conclusion may be linked to the characteristics (length, seasonal patterns, etc.) of the data sets used.

For our analysis, we thus aim to refute or confirm the conclusion obtained from Makridakis et al.'s study as applied to our data set of SKU sales.

2. The Data Set

2.1 Description of the Data

Our raw data set consists of the daily sales, by boxes or packs, of four sizes of a carbonated beverage between January 1st, 2016 and December 31st, 2018. The four sizes are: 1L, 1.5L, 0.5L and 2L respectively. The aim of our forecasting analysis is to predict the sales of the year 2019. Our data set is multivariate in the sense that there are five variables: the date represents the independent variable and the daily sales of the four stock keeping units represent the dependent variables. However, by treating each stock keeping unit separately, we render our data a set of four univariate data sets.

From experience, we can expect our data to contain seasonality since it is expected that the sales of the carbonated beverage increase during summer or whenever temperature rises.

2.2 Data Cleaning using R

Using Excel, we arrange our sales data in columns, and we save our file as a .csv file so that we can import it to RStudio, as depicted in Figure 1.

	Date	1L	3/2	1/2	4/2
2	1/1/2016	246	97	186	194
3	1/2/2016	243	102	199	165
4	1/3/2016	215	230	181	295
5	1/4/2016	0	0	0	0
6	1/5/2016	228	824	227	216
7	1/6/2016	253	93	179	150
8	1/7/2016	263	99	213	206
9	1/8/2016	215	123	182	114
10	1/9/2016	182	62	201	132
11	1/10/2016	232	125	164	141
12	1/11/2016	0	0	0	0
13	1/12/2016	205	117	342	152
14	1/13/2016	281	98	232	158
15	1/14/2016	190	238	227	143
16	1/15/2016	225	134	243	166
17	1/16/2016	166	95	157	91
18	1/17/2016	231	64	192	210
19	1/18/2016	0	0	0	0
20	1/19/2016	209	118	202	166
21	1/20/2016	203	294	218	158
22	1/21/2016	238	113	232	176

For R

Figure 1. Excerpt of the arranged raw data set of the four chosen SKUs

Now, we are ready to import our data set in R. After opening RStudio, the integrated development environment is chosen, this is followed by setting the directory to the location of the file where data is stored (Figure 2).

© IEOM Society International

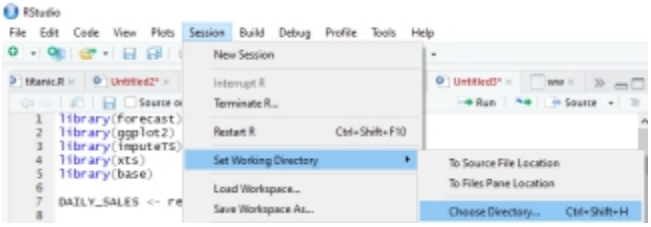

Figure 2. Setting the directory on RStudio

We can now start coding. As shown in Figure 3, we first chose the libraries whose functions we are going to use in order to implement the project (this list of libraries is usually updated as the need for more functions arises further in the coding project). Then, we import the data set from the file that we called "For R.csv" and we store it in the variable DAILY_SALES. On the workspace, the variable is created such that it contains 1096 observations (rows) and 5 variables (columns) including the Date variable, as shown in Figure 4.

Figure 3. Importing our Data Set

	Date	X1L	X3.5L	X0.5L	X2L
1	1/1/2016	246	97	186	194
2	1/2/2016	243	102	199	165
3	1/3/2016	215	230	181	295
4	1/4/2016	0	0	0	0
5	1/5/2016	228	824	227	216
6	1/6/2016	253	93	179	150
7	1/7/2016	263	99	213	206
8	1/8/2016	215	123	182	114
9	1/9/2016	182	62	201	132
10	1/10/2016	232	125	164	141
11	1/11/2016	0	0	0	0
12	1/12/2016	205	117	342	152

Showing 1 to 12 of 1,096 entries

Figure 4. Excerpt of the stored data in RStudio

In the preparation phase of our data for the forecasting analysis, we notice from Figure 4 that some days such as Mondays have a record of zero sales. That is because on most Sundays the sales team is not working, however since the sales process follows a J+1 system (i.e. what is sold today is delivered tomorrow) the sales of Saturdays are recorded on the table under Sundays, and the zero sales on Sundays are recorded under Mondays since no delivery is made on Mondays.

In order to avoid having recurrent outliers in the data, we convert the daily data to weekly data such that our forecasts will be for the weeks of the year 2019. To do that, as a first step, we remove the column of dates in order to only perform the calculations on the data. In the same command line, we convert the data set to a time series using the ts() function. Then, we set the frequency of the data to 7 in order to divide it into weeks. In the workspace in Figure 5, we see that the variable y.ts only has 156 observations corresponding to the 156 weeks in the three years that the data

© IEOM Society International

spans. In order to express the number of weeks in terms of the years, we reset the frequency of our new weekly data set to 52 to assign each 52 weeks to one year.

```
8
9  DAILY_SALES1<- ts(DAILY_SALES[,-1], frequency=365, start = 2016)
10 y.ts <- ts(DAILY_SALES1, frequency=7)
11 WEEKLY_DATA<-aggregate(y.ts, FUN=mean)
12
13 WEEKLY_DATA2<- ts(WEEKLY_DATA, frequency=52, start = 2016)
14
15 autoplot(WEEKLY_DATA2) +
16 ggtitle("Volume of Sales") +
17 xlab("Weeks") +
18 ylab("Number of Packs")
```

```
Data
DAILY_SALES     1096 obs. of 5 variables
DAILY_SALES1    Time-Series [1:1096, 1:4] from 2016 to 2019: 246 24..
WEEKLY_DATA     Time-Series [1:156, 1:4] from 1 to 156: 207 186 182..
WEEKLY_DATA2    Time-Series [1:156, 1:4] from 2016 to 2019: 207 186..
y.ts            Time-Series [1:1096, 1:4] from 1 to 157: 246 243 21..
```

Figure 5. Code for converting daily data into weekly data

Plotting our weekly data set which we arranged in terms of years (2016 to 2019) yields the graph on Figure 6.

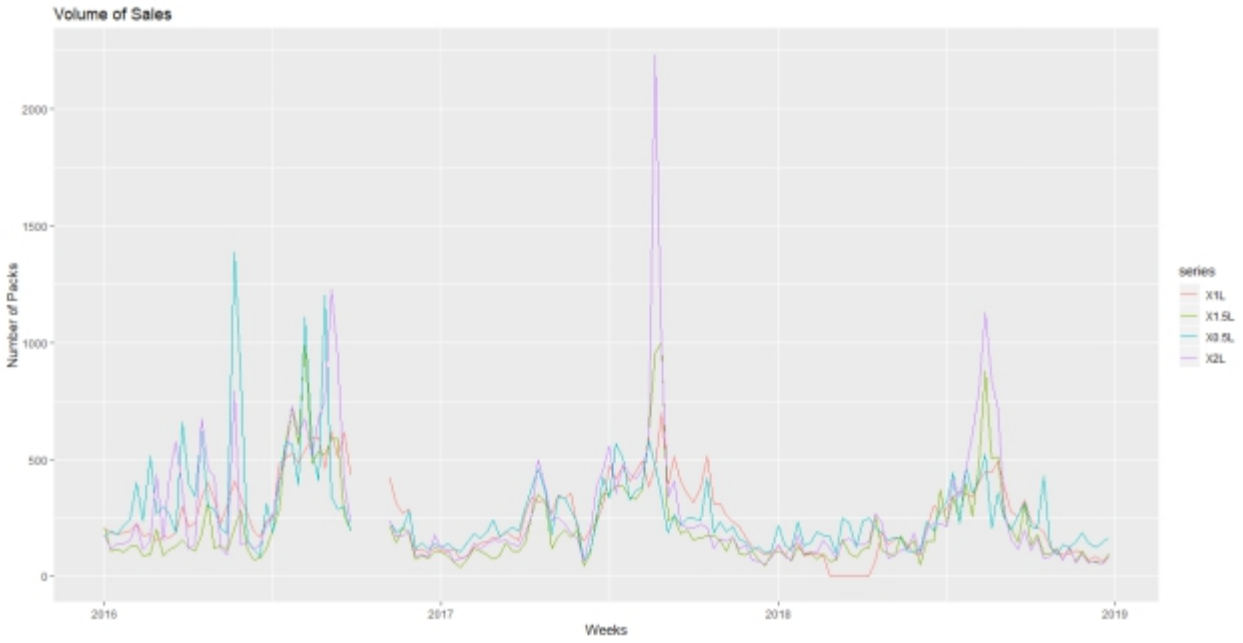

Figure 6. Time plot of sales per SKU

From the graph in Figure 6, we notice that for all four SKUs there is missing data around the end of the year 2016. In fact, the missing observations pertain to the month of October 2016 because the Excel file corresponding to that month got damaged, and thus became unusable. Moreover, we can see that the four data curves follow a seasonal trend such that sales increase during the beginning of the second half of the year (i.e. during summers) with dips around the end and beginning of the year (i.e. during winters). The only exception to the latter observation is the beginning of the year 2016 such that we notice fluctuations in the sales of the all four SKUs alternating between high and low sales. The unusually high sales of the 0.5L SKU around April 2016 may be due to a festivity. The SKU that seems to be sold the most during summers is the 2L bottle, followed by the 1.5L. Conversely, during winters, the most popular SKU among the four in this study is the 0.5L one. In other words, during summer, customers prefer to purchase the family-serve bottle, while during winters, they prefer the single-serve bottle as compared to the others.

The next step in the data cleaning process is to deal with the observed missing observations. To that end and for the sake of simplicity and accuracy, we choose to interpolate the missing data using the na.seadec() function under the imputeTS library as shown in Figure 7.

© IEOM Society International

```
WEEKLY_DATA3 <-na.seadec(WEEKLY_DATA2, algorithm = "interpolation")

autoplot(WEEKLY_DATA3) +
  ggtitle("Volume of Sales") +
  xlab("Weeks") +
  ylab("Number of Packs")
```

DAILY_SALES	1096 obs. of 5 variables
DAILY_SALES1	Time-Series [1:1096, 1:4] from 2016 to 2019: 246 24…
WEEKLY_DATA	Time-Series [1:156, 1:4] from 1 to 156: 207 186 182…
WEEKLY_DATA2	Time-Series [1:156, 1:4] from 2016 to 2019: 207 186…
WEEKLY_DATA3	Time-Series [1:156, 1:4] from 2016 to 2019: 207 186…
y.ts	Time-Series [1:1096, 1:4] from 1 to 157: 246 243 21…

Figure 7. Dealing with missing data using interpolation

After plotting the updated data set with all the data mapped out, we get the graph in Figure 8.

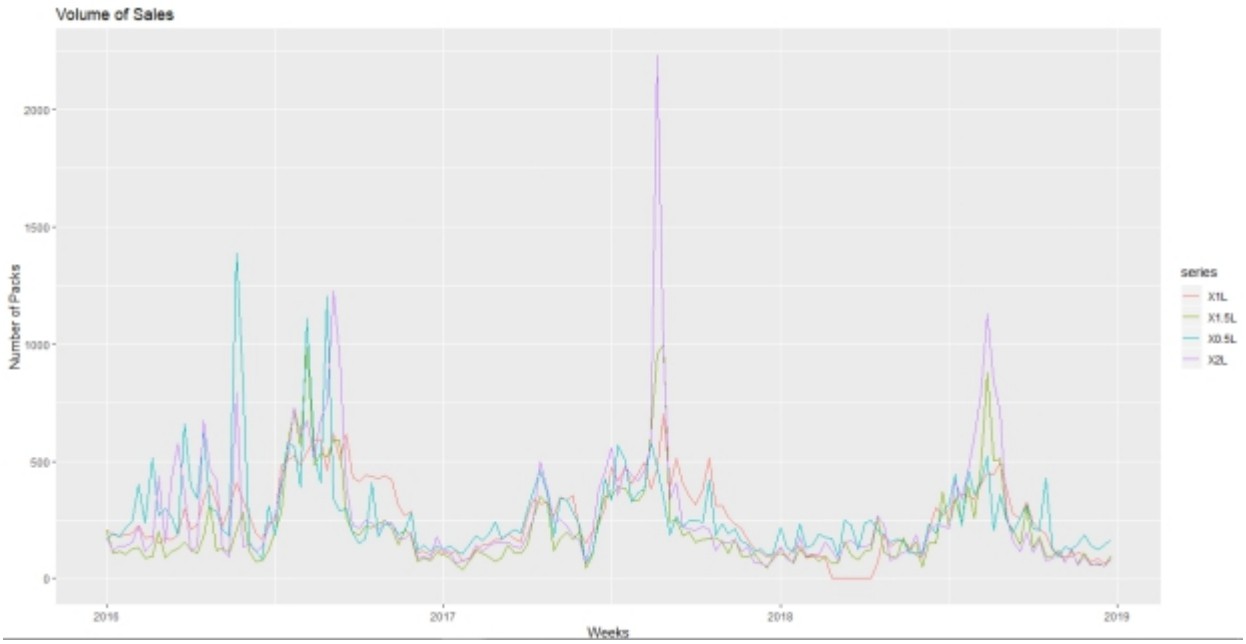

Figure 8. Time plot of complete data of sales per SKU

The replaced data follows the same trends observed in the rest of the data. In other words, the interpolated data decreases in quantity as the end of the year 2016 draws nearer, and it follows the same fluctuations as the data around it for each SKU.

X1L	X1.5L	X0.5L	X2L
Min. : 0.0	Min. : 38.14	Min. : 65.43	Min. : 52.57
1st Qu.:127.5	1st Qu.: 97.25	1st Qu.: 160.32	1st Qu.: 116.14
Median :226.9	Median : 143.71	Median : 217.00	Median : 169.86
Mean :256.2	Mean : 208.56	Mean : 271.59	Mean : 272.83
3rd Qu.:366.4	3rd Qu.: 248.56	3rd Qu.: 333.89	3rd Qu.: 357.61
Max. :702.9	Max. :1000.71	Max. :1386.29	Max. :2233.43

Figure 9. Summary Statistics

Figure 9 depicts the summary statistics of the four quantitative sales variables. We notice that the mean sales of all four SKUs are almost similar. Moreover, the great difference between all maximum and minimum sales values indicate that there must be a seasonality during which sales peak (Max.).

After generating seasonality plots for each SKU using ggseasonplot(), we notice that there is a long-term decrease in the volume of sales from year to year. In order words, the data contains a trend that is not obvious from the graph of Figure 8.

© IEOM Society International

3. Forecasting Analysis

A great amount of research has been conducted in the field of Artificial Intelligence and its applications in forecasting through its subfield Machine Learning (ML) and more specifically Neural Networks (NN). This interest in ML methods in forecasting sparks from the increased accuracy that these methods are claimed to provide, as opposed to classical statistical methods, from the first application of NN in 1964. To that end, we aim at testing this hypothesis by comparing classical and NN forecasting methods on the own data set of weekly sales. The traditional methods we choose are ARIMA, TBATS and ETS coupled with STL decomposition, which we compare to The Neural Networks Auto-Regression in terms of accuracy and minimized error.

3.1 Traditional Methods

Starting off with ARIMA, we forecast each SKU separately as a univariate data set since the function auto.arima() in R doesn't accept multivariate data sets. To that end, we divide the data set into a training and a testing set (104 observations to 52 observations), as shown in Figure 10. Then we use the function auto.arima() that looks for the best model by minimizing the Akaike Information Criterion (AIC) which is a good measure of the deviation from data points outside the sample. Moreover, we add Fourier terms to account for seasonality and we plot the graph in Figure 11 that depicts both the forecast and the test set plot for the 1L SKU.

```
training <- subset(WEEKLY_DATA3, start=1, end=104)
test <- subset(WEEKLY_DATA3, start=105)
frc1 <- auto.arima(training[,1], xreg=fourier(training[,1], K=13), seasonal = TRUE)
new <- fourier(training[,1], K = 13, h = 52)
frc11 <- forecast(frc1, xreg = new)
autoplot(frc11) + autolayer(test[,1])
```

Figure 10. Forecasting using ARIMA and Fourier terms (Code)

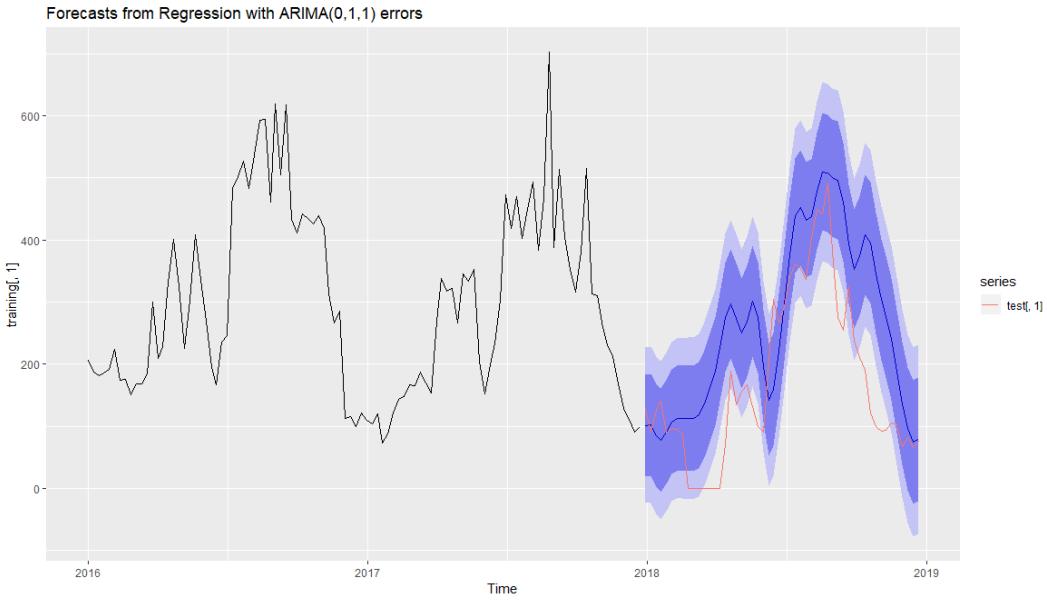

Figure 11. Forecast plot using ARIMA and Fourier terms for the 1L SKU

From Figure 11, we notice that the fitted model is close to the test model. We can thus be confident that the ARIMA model will yield a good accuracy. We thus repeat the process for the remaining three SKUs. For the 1.5L SKU specifically, the forecast fits the testing data set almost perfectly, while it does so accurately for the two remaining SKUs.

© IEOM Society International

We now move on to the TBATS model which is a model that deals well with seasonality and high data frequency (frequency of 52 is deemed high by R). Since this model requires at least three years of data to model seasonality, we directly apply the function tbats() to the whole original data set. Consequently, the forecast will be for the year 2020. The commands in Figure 12 highlight this procedure.

```
datatbats <- tbats(WEEKLY_DATA3[,1])
fc <- forecast(datatbats, h=52)
autoplot(fc, ylab="Number of Packs")
```

Figure 12. Forecasting using TBATS (Code)

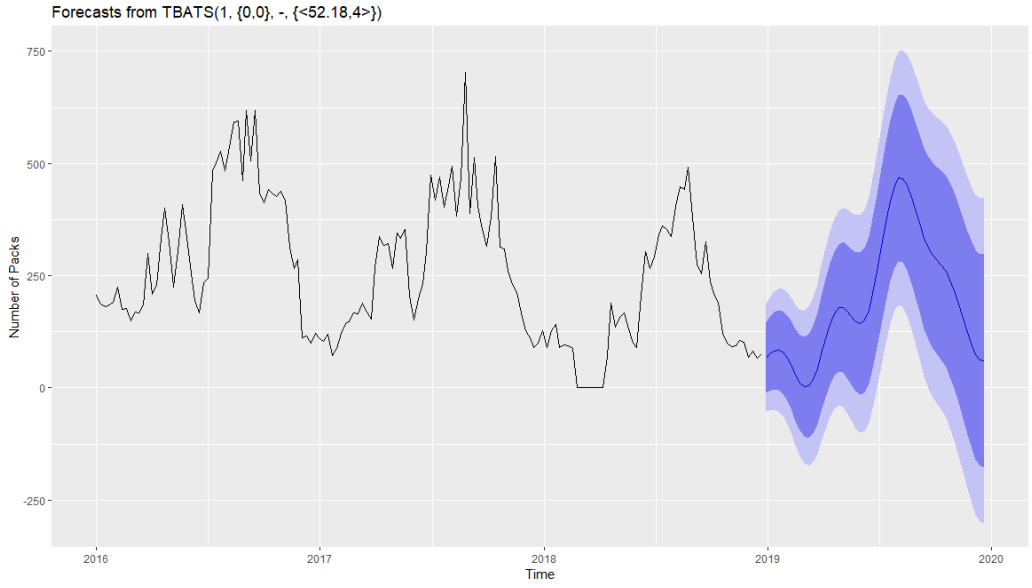

Figure 13. Forecast plot using TBATS for the 1L SKU

From Figure 13, we notice that the forecasted part of the plot is ill-defined and sloppy. Nonetheless, it follows the shape of the previous year's data distribution.

As for the ETS method, we couple it with STL decomposition that distinguishes in the data between the trend component, the seasonality component and the noise or error component. As an example, we apply STL decomposition on the 1L SKU sales. We thus get the plot depicted in Figure 14.

© IEOM Society International

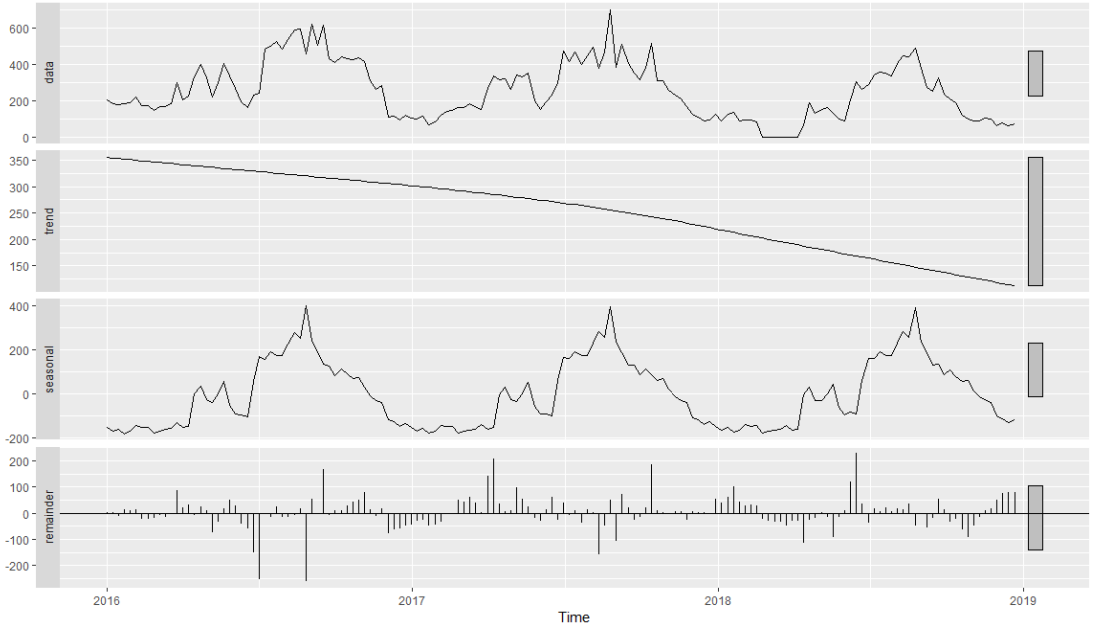

Figure 14. STL decomposition on the sales data of the 1L SKU

From the top, the first plot is the distribution of the original sales data for the 1L SKU. The remaining three plots constitute the breakdown of the first one. The second plot represents the underlying trend of the data; we notice that as the years go by, the sales for this SKU go down. This observation agrees with the results obtained from the seasonal plot of this SKU. The third plot represents the seasonality of the data. As mentioned before, the data contains a yearly seasonality such that sales tend to go up during the beginning second half of the year. Finally, the fourth plot represent the noise or the remainder. A good decomposition means that the remainder should never follow a pattern. Otherwise, the trend and seasonality plot would not have captured all the patterns in the data.

Finally, we now apply an STL forecast that uses ETS on a de-seasonalized version of the sales data set (for the 1L SKU) and re-seasonalizes the forecast. To that end, we use the function stlf() as in Figure 15 and we get the plot in Figure 16.

```
forecast1 <- stlf(WEEKLY_DATA3[,1], t.window=30, s.window=7, robust=TRUE)
autoplot(forecast1)+
  ylab("Number of Packs Sold")
```

Figure 15. Code for STL decomposition coupled with ETS for the 1L SKU

© IEOM Society International

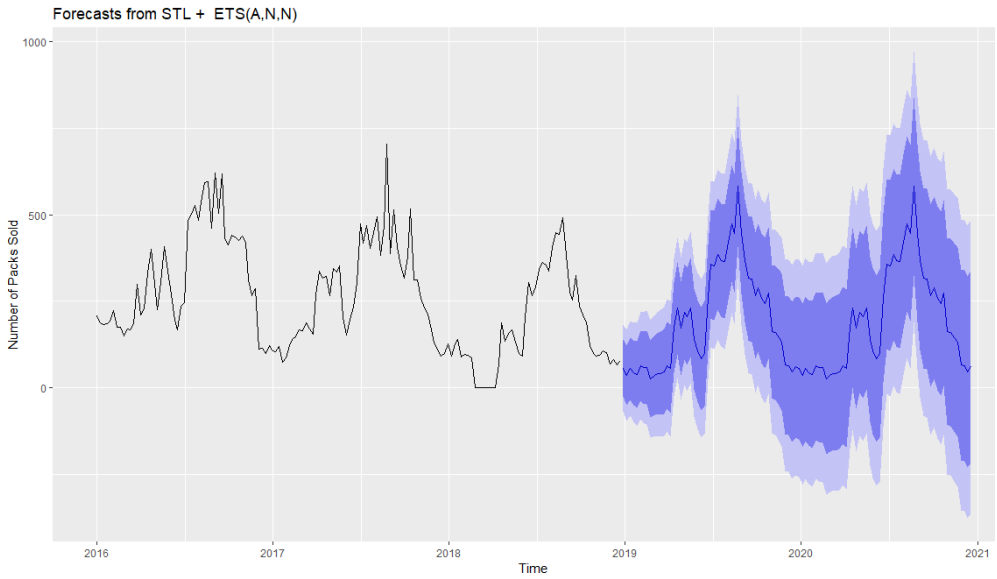

Figure 16. Forecast plot of the weekly sales of the 1L SKU using STL + ETS

The forecast in Figure 16 seems to be representative of the data distribution during the years before 2019. Thus, we may expect the accuracy of this model to be high as well.

To conclude this section, we construct a summary of the accuracies of the three models applied by comparing the AIC and the Mean Absolute Scaled Error (MASE).

Table 1. Comparison of the AIC and MASE for all three models

Approach	AIC	MASE
ARIMA + Fourier Terms	1175	0.501
TBATS	2093	0.458
STL + ETS	2089	0.440

From Table 1, in terms of the AIC, the most accurate model is the ARIMA + Fourier Terms such that this model has the least deviance from out-of-sample data points. However, in terms of the MASE, all three values are less than 1 indicating that the three models are more accurate than the Naïve benchmark (Makridakis et al. 2018). The third model, nevertheless, has the lowest error and we may consider it to be the most accurate event though all three scores are close in value.

3.2 Forecasting using Neural Networks

Neural Networks (NNs) are mathematical models that have been developed and used for ensuring a higher accuracy in several tasks. Basically, an NN is a fully interconnected network made of multiple layers, each made of several nodes, the nodes play an important role, they help build and transform the inputs into outputs Shiffman (2012), Karray, Golhani (2004). The connection between one neuron in one layer to another neuron in another layer has a weight value associated with it. During the prediction phase, the weights are used as coefficients that scale (amplify or minimize) the input signal to a given neuron in the network. The strength of Neural Networks lies in modelling complex nonlinear relationships between some dependent and independent variables (Hyndman and Athanasopoulos 2018). Each layer receives the inputs from the previous layer that (except the input layer itself), performs a nonlinear transformation and moves it down to the next layer as shown in Figure 17.

The processing ability of the network is stored in the inter-unit connection weights, obtained by a process of learning from a set of training patterns. Learning is the process of transforming the connected weights between nodes as a result

© IEOM Society International

of the deviation between the targeted output and the predicted output of such NN in response to the input information at the input layer. There are many techniques commonly used in the learning algorithms which include Gradient Decent Backpropagation, Radial Basis Function and the fastest known technique called Levenberg –Marquardt (LM).

Initially the weights are generated randomly, and a cost function is used to measure the discrepancy between the prediction and the ground truths or observed data. During training, the backpropagation phase usually uses Gradient Descent to update the parameters of the network (weights and biases), and hence to decrease the loss (error). Then information simulates with specific values stored in those weights that enable the networks to have capabilities like learning, generalization, imagination and creating the relationship within the network Tarassenko (1998).

A simple NN has a bottom layer (input or predictors) and a top layer (output or forecast). Most of the economic value of Neural Networks comes from Supervised Learning tasks, namely classification and regression. In our case, and considering the numerical nature of the target output, we opted for building a regression model. Figure 17 shows an example of a shallow NN with one hidden layer having 3 nodes, besides the input and output layers.

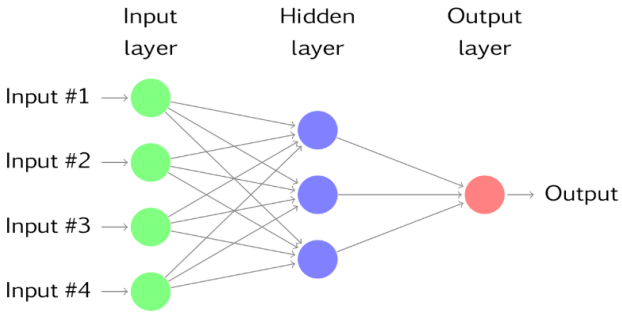

Figure 17. Neural Network with four inputs and one hidden layer with nodes (Hyndman and Athanasopoulos 2018)

Golhani et al. (2018) provide a well detailed description of the most known types and models of Neural Networks. He classified models into Feed-Forward Neural Networks (FFNNs) and Back-Propagation Neural Networks (BPNN). FFNNs use the forward direction in their transformation network. They are mostly used in non-parametric data analysis and are perceived as a good alternative to classic pattern classification and clustering techniques. The non-linear activation function uses Logistic function for hidden nodes and Softmax function for output nodes for multi-class classification tasks.

The Back-Propagation Neural Networks (BPNN) information is directed at the output layer, as opposed to FFNN where information is fed from the input layer to output layer. The author provides a description of the major types of NNs which are Single-Layer Perceptron (SLP), and Multi-Layer Perceptron (MLP), these types vary in terms of the hidden layers existing between the input and the output layers. Among NNs other types exist: Radial-Basis Function (RBF) networks, Kohonen's Self-Organising Map (SOM) networks, Probabilistic Neural Network (PNN), and Convolutional Neural Network (CNN).

Data Pre-processing

In order to train the NN, the data collected is divided into training and testing data. Data is commonly divided into 50/50 or 60/40 or 70/30 in terms of percentage of training and testing purposes in the simulation work. Typically, the partition chosen is the one giving the highest coefficient of correlation, R (Arsad et al. 2013).

The research work of Tahiti (2017) answers the question whether NN models provide more accurate results with data preprocessing. To this end, they compared six real time series data, which have trend and seasonal variations. The results indicate that NNs can model and forecast trend and seasonal time series. However, data preprocessing can provide more accurate results. While sometimes a trend adjustment provides better results, sometimes a seasonal adjustment gives better results (Tahiri 2017).

© IEOM Society International

NN Model Design and Selection

To choose the most adequate NN design, we look for the minimization of the deviation between the targeted and the predicted value of the output. There is no unified rule of choosing the parametrization rates, namely the learning rate, the momentum rate and the number of hidden layers in any network. It is rather trial and error process and dependent on the designer of the network. However, there are some heuristics from previous research that can lead to optimized models, e.g., the Learning Rate (0-1) helps to converge the NN training process while the Momentum Rate (0-1) helps to accelerate the training process in the simulation.

We applied the chosen NN model, The Neural Network Auto-Regression (NNAR), on the data set of sales of the 1L SKU. Decision taken due to its large similarity with ARIMA in data processing before the forecasting phase, and its high accuracy relative to machine learning. Sena and Nagwawni (2016) used NNAR in forecasting the quarterly per capita disposable income of West Germany. The used method is articulated around 6 steps:

1. Exploring the major variability in the data.
2. Partitioning data into training and validation sets.
3. A layout of the NN is created by considering the number dependent lags for input and number of hidden layers.
4. Training patterns are prepared for the training of neural network.
5. NN is trained using the training patterns.
6. NN is tested by calculating the mean relative error of forecasting.

NNAR is a feedforward neural network with one hidden layer with p lagged variables. With time series data, lagged values of the time series can be used as inputs to the neural network. This can be achieved via nnetar(), that is, neural network auto regression. nnetar(p,P,0)m is equivalent to ARIMA(p,0,0) (P,0,0)m where m is the frequency parameter, p denotes the number of lagged values and k stands for the number of nodes in hidden layer. With seasonal data, it is often useful to add the last observed values from the same season as inputs. The default values are P=1 and p is chosen from the optimal linear model fitted to the seasonally adjusted data. k is set to (p+P+1)/2. For non-seasonal time series, the default is the optimal number of lags (according to the Akaike information criterion (AIC) for a linear Autoregression model.

For that purpose, we use the function nnetar() in R with no Box-Cox transformation. We thus get the plot in Figure 18.

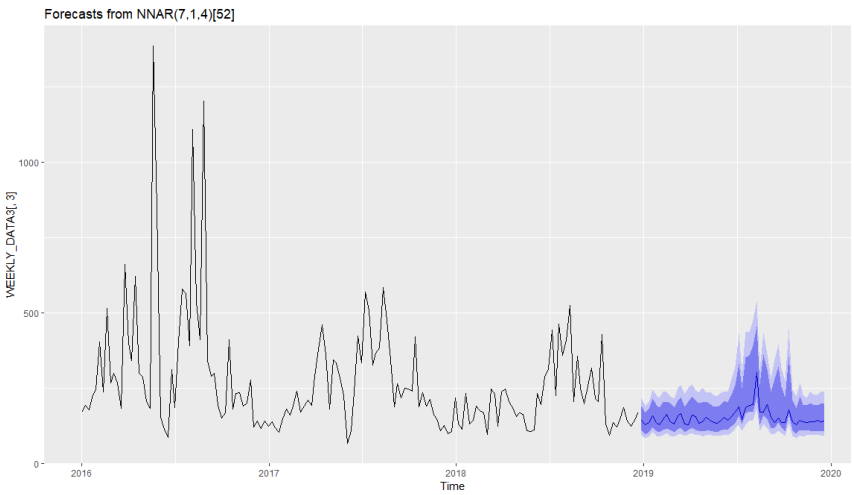

Figure 18. Forecast plot for the 1L SKU using Neural Network Autoregression

From the plot, we notice that the forecasted part follows the decreasing trend of sales as well as its seasonality. Looking at the accuracy of this method, we find that the MASE is 0.23. Additionally, other measures of error are found to be

© IEOM Society International

lower for this method compared to the classical methods. Thus, it is safe to say that this Neural Network Autoregression method is the most accurate and thus the best among the four explored.

4. Conclusion

The aim behind this limited comparative study is purely exploratory. To that end, we performed three classical forecasting methods, namely ARIMA (+ Fourier Terms), TBATS and ETS (+ STL), and one Machine Learning method which is Neural Network Autoregression, on the sales data of the 1L SKU. After plotting and comparing the measures of accuracy of the four methods (using MASE as a metric), we concluded that for the weekly sales data of the first SKU, the Neural Network Autoregression method yields more accuracy.

Future work perspectives include exploring Recurrent Neural Networks and autoregressive models in deep learning, which are sequence models with multiple hidden layers, capable of extracting more features, and which might be useful for predicting times series.

References

Arsad, P. M., Buniyamin, N., and Ab Manan, J., A neural network students' performance prediction model, *IEEE International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA)*, 26-27 November 2013.

Golhani, K., Balasundram, S. K., Vadamalai, G., and Pradhan, B., A review of neural networks in plant disease detection using hyperspectral data, *Information Processing in Agriculture,* vol. 5, no.3, pp. 354-371, September 2018.

Hyndman, R.J., and Athanasopoulos, G., *Forecasting: Principles and practice*, 2nd edition, OTexts: Melbourne, Australia, OTexts.com/fpp2, 2018.

Karray, F. O., and De Silva, C., *Soft computing and intelligent systems design: theory, tools, and applications*, London: Pearson Education, 2004.

Makridakis, S., and Spiliotis, E., and Assimakopoulos, V., Statistical and machine learning forecasting methods: Concerns and ways forward*, PLOS ONE,* vol. 13, no. 3, 2018.

Sena, D., and Nagwani, N. K., A neural network autoregression model to forecast per capita disposable income, *ARPN Journal of Engineering and Applied Sciences,* vol. 11, no. 22, November 2016.

Shiffman, D., *Neural networks*. The nature of code: simulating natural systems with processing (Online), Link: http://natureofcode.com/book/chapter-10-neuralnetworks/, 2012.

Tarassenko, L., Neural computing hardware and software*,* In*: Guide to neural computing applications,* Butterworth-Heinemann, pp. 59–66, 1998.

Tarihi, G., and Tarihi, K., Neural network data preprocessing: is it necessary for time series forecasting?, *Journal of Academic Researches and Studies,* vol. 9, no. 17, pp. 147-154, 2017.

© IEOM Society International

Biographies

Ghita Benboubker is a senior undergraduate student enrolled at Al Akhawayn University of Ifran, Morocco. She is majoring in Engineering and Management Science with a concentration in Operations Research. Aside from her academic occupations, she currently serves as the University Honors Program Student Representative, as the Mechanics of Materials Student Representative and as a Resident Assistant at her university. Her main areas of interest include decision support systems and optimization. The present paper is based on one of her internship experiences.

Dr. Ilham Kissani is an assistant professor in the field of engineering management for the School of Science and Engineering at Al Akhawayn University in Ifran, Morocco. She has served as the main advisor and lead instructor for the undergraduate and MS programs in Engineering and Management Systems since 2010. She has helped create very close ties with the AUI School of Business Administration, which allows both schools to leverage their resources and deliver a greater diversity of courses to students, such as supply chain management and operations management. Her background is diverse and includes industrial as well as academic experience. Her degrees are from INSEA, Morocco (Engineer) and Université Laval, Canada (Master and Ph. D). She has worked with Royal Dutch Shell as a project manager and with Modellium Québec, where she provided consulting services in logistics and supply chain issues in Canada and Brazil. Additionally, Dr. Kissani contributes in research in supply chain management, planning, and operations research. She is a member of ASEM, IEEE, IEOM, IIE, and INFORMS.

Dr. Asmaa Mourhir holds a Ph. D in Computer Science from the University of Sidi Mohammed Ben Abdellah, Fez. Currently she holds a position as Assistant Professor of Computer Science at Al Akhawayn University in Ifran, where she has been serving since 2003, and contributed to a number of research and development projects (Project funded by Maroc Telecom: Development of a multimedia platform for sponsor advertising over IP/SIP telephony networks; Project funded by Al Akhawayn University: Development of a multi-criteria environmental decision support using an expert system). Professor Mourhir was an active member of the Communications & Systems laboratory at Al Akhawayn University. She is currently a permanent member of LIIAN laboratory (Laboratoire Informatique, Imagerie & Analyse Numérique) at Faculté des Sciences Dhar El Mahraz, Fez. Prior to that, she worked in industry as a software engineer and project manager. Her previous research interests include Voice over IP and Multimedia Processing. Her current research interests are sustainability and environmental modeling, integrated assessment, decision support systems, precision agriculture, expert systems, soft computing, and deep learning.

© IEOM Society International