UIMP
Universidad Internacional
Menéndez Pelayo

Aplicación de herramientas de ML a la predicción de tendencias en función del análisis de históricos de ventas B2B

(Finding Trends in the B2B Textile Sector using Machine Learning)

Trabajo de Fin de Máster
para acceder al

MÁSTER INTERUNIVERSITARIO EN CIENCIA DE DATOS

Autor: Javier Sáez Martínez

Director\es: Steven Van Vaerenbergh
Juan Marcos Sanz Casado

Julio - 2021

ABSTRACT

Machine learning has untapped potential in many branches of knowledge. Moreover, this potential can be used in applications of interest to companies in different sectors. This paper discusses the applications of Artificial Intelligence in demand forecasting in the B2B textile sector. This project is coordinated and developed by Textil Santanderina, one of the leaders in the sector at national level in Spain.

Therefore, this Master's thesis in collaboration with the mentioned firm is focus on the prediction of time series with different Machine Learning techniques, and analyses the potential of the results with the aim of integrating these techniques in the company's decision-making process.

Keywords: Time Series, Supervised Learning, Machine Learning

RESUMEN

El Machine Learning tiene un gran potencial por explotar en muchas ramas del conocimiento. Además, ese potencial puede ser utilizado en aplicaciones de interés para empresas de diferentes sectores. En este *paper* se realiza una discusión sobre las aplicaciones de la Inteligencia Artificial en la predicción de demanda del sector B2B textil. Este proyecto es coordinado y desarrollado por Textil Santanderina, una de las corporaciones líderes del sector a nivel nacional en España.

Por lo tanto, este trabajo de Fin de Máster en colaboración dicha empresa. Se centrará en la predicción de series temporales con diferentes técnicas de Machine Learning, y analizará el potencial de los resultados con el objetivo de integrar estas técnicas en el proceso de toma de decisiones de la compañía.

Palabras clave: Series Temporales, Aprendizaje Supervisado, Machine Learning

Contents

List of Acronyms

MA	Moving Average
AR	Auto-Regressive
ARIMA	Auto-Regressive Integrated Moving Average
BB	Black Box
DL	Deep Learning
DSS	Decision Support System
DT	Decision Tree
EMA	Exponential Moving Average
EML	Ensemble Machine Learning
ES	Exponential Smoothing
GP	Gaussian Process
GPR	Gaussian Process Regression
HDT	Holt's Damped Trend
HLS	Holt's Linear Smoothing
HWSM	Holt-Winters' Seasonal Method
IA	Artificial Intelligence
ML	Machine Learning
MSE	Mean Square Error
NN	Neural Network
RF	Random Forest
RF	Regression Tree
RMSE	Root Mean Square Error
RSS	Residual Sum of Squares
TS	Textil Santanderina

Links

GitHub	https://github.com/javi-saez/TFM-Textil-Santanderina
ZENODO	10.5281/zenodo.5110379

1 Introduction

This research project deals with machine learning in business digitization. The whole process was under the supervision and the data provided by the company Textil Santanderina SA (TS). TS is a Cantabria-based B2B textile company that operates in a wide range of fabric areas. They operate in markets with a cyclical component such as the fashion market.

Behind the volume of data produced by mid-large companies, there are many options to find optimization processes, which is mainly what we know as business digitization [37].Which is the case of TS. According to most recent published numbers. This project deals with the prediction of demand time series based on historical records.[1]

Before the state of art is offered to the reader. It is convenient to explain shortly three main basic concepts for a better understanding of the challenge which tries to overcome this research study. These three concepts are:

1. **Machine Learning**: covers the main discussion in this research project.
2. **Time series prediction**: Forecasting as a machine learning problem.
3. **Business Intelligence**: understood as the process of applying ML as decision support to businesses decisions.

1.1 Concept of Machine Learning

Machine Learning (ML) is a sub-field of Artificial Intelligence (AI)[2] which uses arithmetical approaches to teach a system, thus, a machine, to create knowledge from experience. [2]

This field of study describes the development and implementation of algorithms based on empirical data with the goal of optimization of the results and the improvement of predictions.

This algorithmic implementation takes place with a **supervised** and **unsupervised** learning [37]. Figure 1 illustrates the different attributes and goals in both learning processes.

[1]But note that there are many other uses of automatic learning in business optimization such as customer segmentation, quality control, etc. Nevertheless, this Master's Thesis will focus on one specific issue, and the other uses of Machine Learning are outside of its scope.

[2]Even though, AI has more than one formal definition in literature, in the context of this paper it is understood as "the theory and development of computer systems able to perform tasks normally requiring human intelligence" [49].

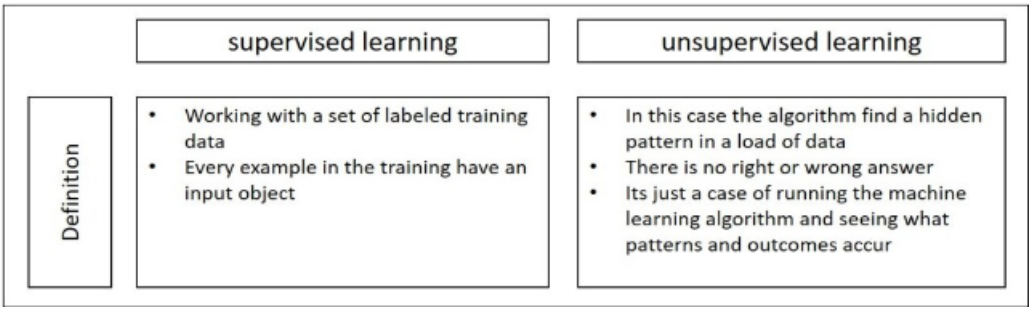

Figure 1: **Main ideas about monitored and unsupervised learning** [2].

1.2 Time-series prediction

In our case, we will have a Machine Learning problem with labeled data. Therefore, it will be possible to compare results according to goodness-of-fit measures. As previously announced, these experiments will try to predict time series. **First of all, what are time series and what characterizes them?**

A *time-series* is a collection of values registered sequentially over time. Many examples could come to anyone's mind quickly: the temperature registered in a concrete point in space, the number of cases reported of an illness in a country,etc. In several areas, a good prediction of these values for the future is essential[3].

As far as the principal part of the discussion is going to be about these procedures, it has been considered important to have a formal definition of *time-series.*

"A time-series is a process in which a vector $X = x(1), x(2), ..., x(n)$*, where each element* $x(t) \in R^m$ *pertaining to* X *is an array of* m *values such that* $x(t)_1, x(t)_2, ..., x(t)_m$*. Each one of the* m *values correspond to the input variables measured in the time-series."* [31]

1.2.1 Time-series patterns

Mostly, one or more **patterns** could be observed in *time-series* data. These structures are helpful in modeling and describing processes of this kind of stochastic processes [19]:

[3]It is also the case in the manufacturing sector, where a good estimate of future demanded quantities could help to optimize stock storing and to moderate production stress in peak moments. But, the reasons for developing this project will be better explained later.

Trend

A trend occurs when there is **a long term increase or decrease** in the data[4]. For instance, a positive trend can be seen in the electricity demand data in Australia (due to economic development and other factors), see Fig. 2.

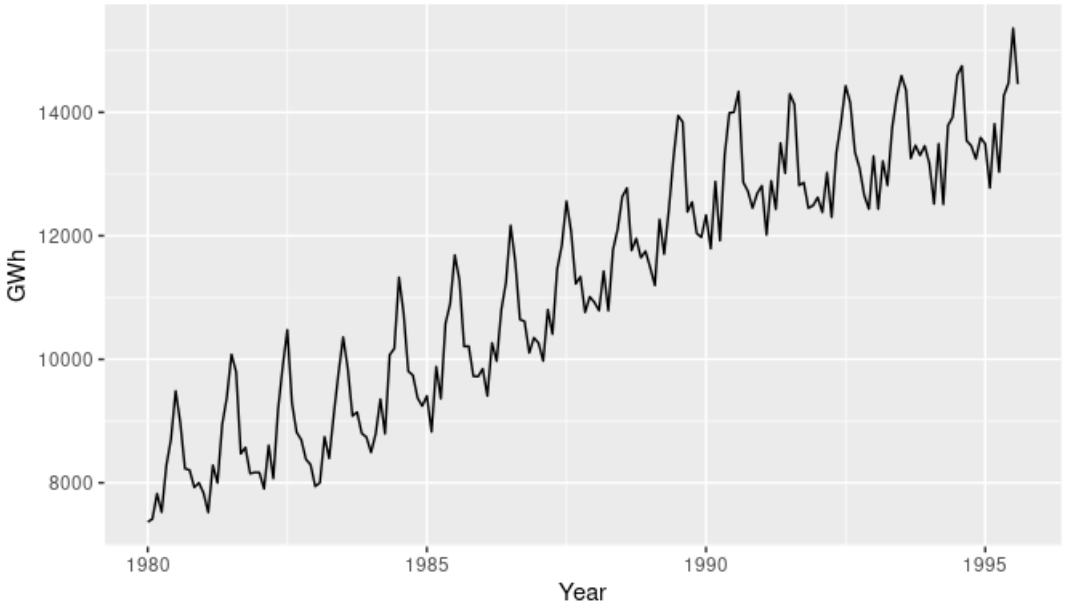

Figure 2: **Energy power demand through time in Australia, plot by** [19]

Seasonality

A *seasonal* order happens when our record has been affected by factors related to the month of the day of the week. To give an example, just let's look on our previous plot, there are peaks and valleys inside each year.

Cyclical patterns

A *cycle* occurs when the evolution of the plot shows increases and falls that are not of a fixed frequency[5]. In this way, these ups and downs are connected to external factors such as economic conditions or the *business cycle.*

The discussion about how to use this sort of sequences for modelling future values [6] will be part of the methodology section of this thesis.

[4]It is not necessarily linear.
[5]Unlike seasonal cases.
[6]So far, just plot interpretation methods have been explained.

1.3 What is Business Intelligence?

"Business intelligence is the process of taking large amounts of data, analyzing that data, and presenting a high-level set of reports that condense the essence of that data into the basis of business actions, enabling management to make fundamental daily business decisions." [55]

There are different definitions within academia, but the latter one provided by Stackowiak et al. is particularly appropriate for the context of this research paper.

Data science, and therefore Machine Learning, has brought companies from all sectors the opportunity to take business efficiency to the next level. This has a huge impact on tasks such as interpreting future risks, predicting demand, and making better strategic decisions, to name a few examples.

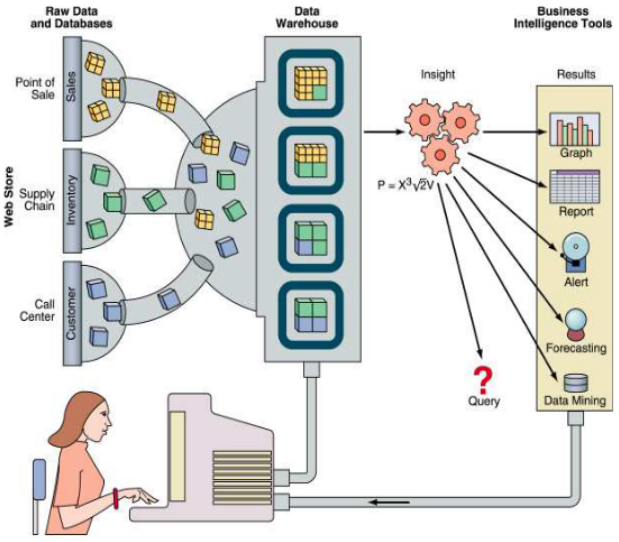

Figure 3: **Basic understanding of the BI process** [44]

This project tries to show the benefits of BI, with the assistance of ML, with respect to the usual expert intuition.

1.4 Motivations and scope of this project.

Finally, a brief elaboration of the motivations and applicability of this project in the future will be presented:

When optimal decisions are made, more plausible would be for a firm to persist over time. The COVID-19 outbreak is likely to cause bankruptcy for many well-known brands in many industries as consumers stay at home and economies are shut down. Then, many industries have been left in a very difficult situation worldwide [11].

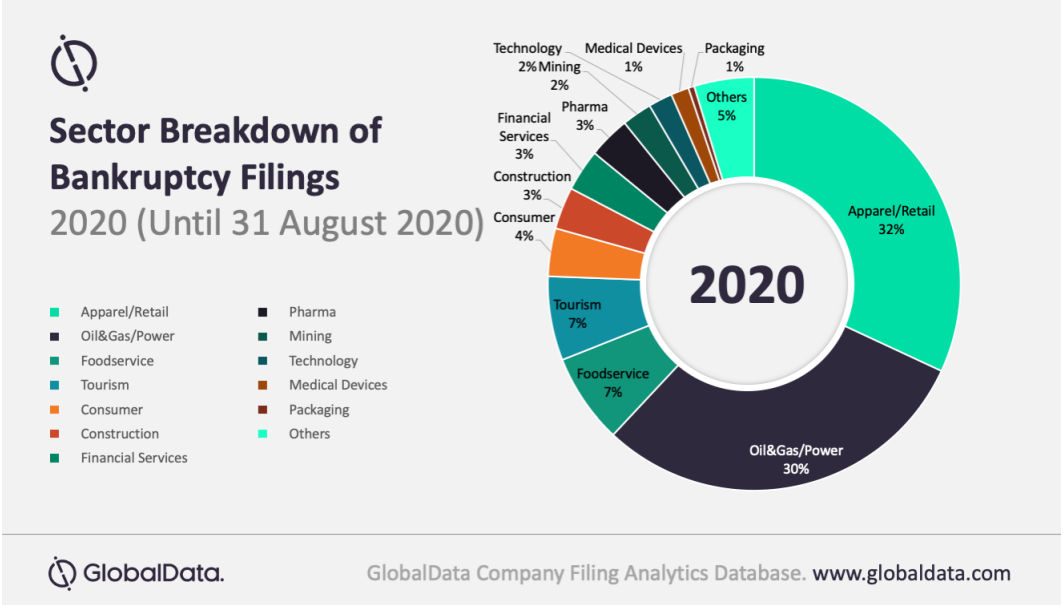

Figure 4: **It is manifest, that essentially, the impact of the COVID-19 has been suffered by most cyclical sectors. Source: Globaldata.**

In these circumstances, it has been seen in scientific literature papers related to Machine Learning and demand prediction in the COVID-19 pandemic.[7] Therefore, there is an impact concerning the current moment, enterprise efficiencies are needed more than ever. Even without a pandemic, digitization will continue to be necessary in many business areas in order to gain competitiveness. It may be that the current situation has been a final impetus for this transformation.

[7]There is an interesting example, published by the journal Nature Energy and developed by several researchers [2]. The authors of this paper achieved with success combustible demand prediction during the pandemic in the US based on human mobility. This article can be found in the bibliography of this document.

2 State of the art

As has been previously explained, the ML experiments tested in this project will be related to time-series prediction. Then, concerning the purpose of this project in analyzing the current panorama in the ML field, **the State of the Art of this thesis will be dedicated to the most commonly used algorithms in time-series prediction nowadays.** Also, recent trends are going to be analyzed.

Although the interest of this state of the art is to give an overview, it is far from exhaustive. Indeed, a rich body of literature exists surrounding traditional ML methods such as kernel regression [35] and support vector regression [52].

2.1 Exponential Smoothing and ARIMA

Even though these two methods seem pretty classic, they appear in recent research papers with satisfactory results. In the following paragraphs, the major technical differences between the two models will be discussed. In addition, real-life cases will be presented in which these solutions have been particularly attractive to researchers from different disciplines.

2.1.1 Exponential Smoothing model

Exponential smoothing (ES) was first proposed as a predictive solution in the late 1950s [5] or early 1960s [60]. The idea behind this method is very simple: **predictions are made using weighted averages of past records, with the weights falling exponentially as the observations are farther in the past** [19]. There are different variants of this approach, the choice of each of which is based on the components the researcher observes in the time series[8] and the way in which the smoothing effect is to be applied[9].

Simple Exponential Smoothing

This method is suitable when no trend or seasonality is apparent in the process to be predicted. The math behind this is quite simple, the model will have only one parameter to optimize (α). Note that when optimising a parameter [10] based on an error metric [11], no matter how simple the algorithm is, it may be already one step ahead of simple human intuition.

[8] Trend and seasonality.

[9] Mostly additive errors, damped, multiplicative manner, and so on. The most used ones will be explained latter.

[10] The parameter's value could be optimized using different approaches. For example, minimizing the sum of squared errors.

[11] Different options for this type of problem will be proposed to the reader in the methodology.

$$\hat{y}_{T+1|T} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots \tag{1}$$

As shown in Eq. (1) and in the Fig. 5, the lower the value of alpha, the less consideration is given to the past values of the target variable. This method is quite useful when challenges with few data are faced [12], and the specific time-series has no seasonality or trend [19].

	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$
y_T	0.2000	0.4000	0.6000	0.8000
y_{T-1}	0.1600	0.2400	0.2400	0.1600
y_{T-2}	0.1280	0.1440	0.0960	0.0320
y_{T-3}	0.1024	0.0864	0.0384	0.0064
y_{T-4}	0.0819	0.0518	0.0154	0.0013
y_{T-5}	0.0655	0.0311	0.0061	0.0003

Figure 5: **Value contributed in the prediction by the different records according to alpha.**

There are adaptations of this approach to include different components of the time series in the ES model[19]:

1. **Holt's Linear Smoothing** (HLS): On the one hand, Holt [18] also took into account the trend component within these processes with the following variation of the ES framework. Now the ES model has two factors to take into account (and optimise) when a number h of observations in the future need to be predicted: α referring to the level component and β referring to the trend [13]. Note also that this alternative is only recommended if we observe a trend in the historical data, but no seasonality.

$$\text{Forecast equation} \quad \hat{y}_{t+h|t} = \ell_t + hb_t \tag{2a}$$

$$\text{Level equation} \quad \ell_t = \alpha y_t + (1-\alpha)\left(\ell_{t-1} + b_{t-1}\right) \tag{2b}$$

$$\text{Trend equation} \quad b_t = \beta^*\left(\ell_t - \ell_{t-1}\right) + \left(1-\beta^*\right) b_{t-1} \tag{2c}$$

[12] This is the case in the ML experiments developed in this project.

[13] As can be noted by the reader, by adding more parameters, we make the model more complex with respect to human intuition.

2. **Holt's Damped Trend** (HDT): The problem with the linear Holt variant to the simple ES is that the trend remains constant for all forecasts in the future, due to this fact HLS tends to overestimate the effect of this trend in the long run. Therefore, HDT includes a third damper ϕ parameter[14] in order to compensate for this overestimation of the trend as it is shown in Eq. (3).

$$\text{Forecast equation}^* \quad \hat{y}_{t+h|t} = \ell_t + \left(\phi + \phi^2 + \cdots + \phi^h\right) b_t \tag{3a}$$

$$\text{Level equation}^* \quad \ell_t = \alpha y_t + (1-\alpha)\left(\ell_{t-1} + \phi b_{t-1}\right) \tag{3b}$$

$$\text{Trend equation}^* \quad b_t = \beta^*\left(\ell_t - \ell_{t-1}\right) + \left(1-\beta^*\right)\phi b_{t-1} \tag{3c}$$

In the long run our predictions will become constant when $0\phi < 1$ since the limit as h$\to\infty$ [15] in the forecast function Eq. (3a) is $\ell_T + \phi b_T/(1-\phi)$. **This indicates that predictions in the short run will have that trend component while those further out will be constant**. A very intuitive example is given in Fig.6.

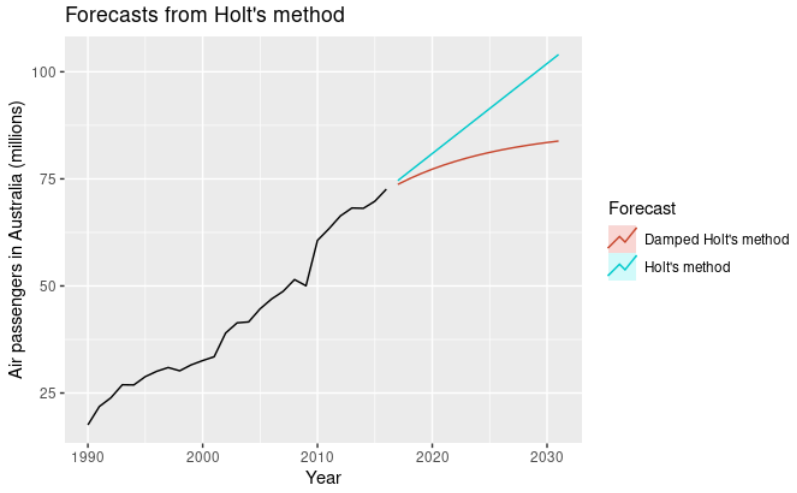

Figure 6: **Long run differences between HLS and HDT [19].**

3. **Holt-Winters' Seasonal Method** (HWSM): Holt (1957) and Winters (1960) extended last mentioned methods to capture the seasonality component in the time series in a new approach. In this scheme there is an additional equation: the seasonal component, s_t, In this new function we will find a new parameter representing the seasonality of the time series, let's call this variable gamma (γ) [18] [60].

[14] In practice, the optimal values of ϕ are not less than 0.8 (and not greater than 1) [19].

[15] Where the variable h is understood as the number of predictions to be made since the last record.

$$\text{Forecast equation}^{**} \quad \hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)} \tag{4a}$$

$$\text{Level equation}^{**} \quad \ell_t = \alpha\,(y_t - s_{t-m}) + (1-\alpha)\,(\ell_{t-1} + b_{t-1}) \tag{4b}$$

$$\text{Trend equation}^{**} \quad b_t = \beta^*\,(\ell_t - \ell_{t-1}) + (1-\beta^*)\,b_{t-1} \tag{4c}$$

$$\text{Seasonality equation} \quad s_t = \gamma\,(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m} \tag{4d}$$

In this framework, m would be the number of units into which we divide the year ($m = 12$ for months or $m = 4$ for quarters, for example). Furthermore, k is the integer result of (h-1)/m, so the cyclical component used in the prediction is the final year of our training sample. Notice in (4d) that the seasonality equation is a weighted average between the current time index $(y_t - \ell_{t-1} - b_{t-1})$ and the same from the same observation one year before [19]. Then, this effect would be added for each observation in our forecast equation, this approach works good when the seasonal effect is constant among time [16].

As already mentioned in the preamble of this section, ES models have a current presence in the scientific literature when solving time series forecasting problems. As already mentioned in the preamble of this section, ES models have a current presence in the scientific literature when solving time series forecasting problems. A relatively recent example is the paper published by Duan Niu R. (2018) in which they used ES methods to predict the surface loss of different lakes in Wuhan [12]. In this case, given the limited availability of data, ES techniques are recommended. Note also Fig. 7.

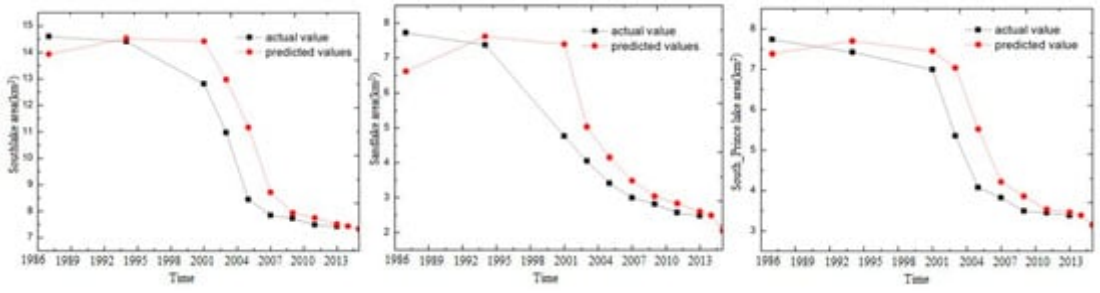

Figure 7: **Lake dimension prediction in Wuhan using ES scheme** [12]

2.1.2 ARIMA frameworks

ARIMA models provide another approach to the prediction of these so-called time series processes. ES and ARIMA are the most openly used methods for this type of problem, moreover, both schemes complement each other: while ES focuses on parameterising the trend and/or seasonality of the series, ARIMA frameworks aim to model the auto-correlation in the data. [19].

[16] There is another possibility for the additive method, instead of adding up the seasonal component, it would be advisable to multiply it. This last alternative is more desirable when the seasonal component is not constant from year to year.

In order to understand how the ARIMA model works roughly, its main components **(p, d, q)** will be explained in the following key points: [17].

1. **Autoregressive component, p** (AR): in an AR model, the target variable is predicted by using past values of itself as a regressor. Thus, the autocorrelations of that same value:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t \tag{5}$$

   In this case ε_t (the component's residual) is white noise [18]. As can be seen, this element from ARIMA is like a simple/multiple regression but with past values of y as predictors. An $AR(p)$ model takes considers delays of the target variable of order p.
2. **Integrated component, d** (I): from Eq. (5), the parameter c refers to the mean value of the differences between past observations (if c is positive the trend of the series will be of this form and conversely). These differences are used to eliminate the trend component in the series [20].
3. **Moving average component , q** (MA): instead of using delayed values of y_t a moving average model uses past forecast residuals [19]. Note in Eq. (6) than a MA(q) model is a moving average model of order q.

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \tag{6}$$

After combining the three components mentioned above, an **ARIMA(p,d,q) model** is obtained.

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{7}$$

Nevertheless, this model needs a seasonal component, **s**, when the data has this type of feature. In that case the ARIMA workspace becomes SARIMA [19]. In these SARIMA models an additional seasonal term is included, where m is the number of annual records in the time series to be predicted.

$$\text{ARIMA } \underbrace{(p, d, q)}_{\uparrow} \quad \underbrace{(P, D, Q)_m}_{\uparrow}$$

Non-seasonal part — Seasonal part of of the model of the model

[17] There are details within this approach that require a much more technical and detailed explanation, this part of the literature review relies on these two resources: [19] and [20], same case with ES models.

[18] Time series that have no autocorrelation are known as white noise.

As in the case of the ES schemes, ARIMA and SARIMA models are currently used for predictions on issues of importance, as was also done during the COVID-19 pandemic when there was a large autocorrelation between one week's and the previous week's infection data in the pandemic. This type of estimations allowed the responsible authorities to make life-saving decisions.

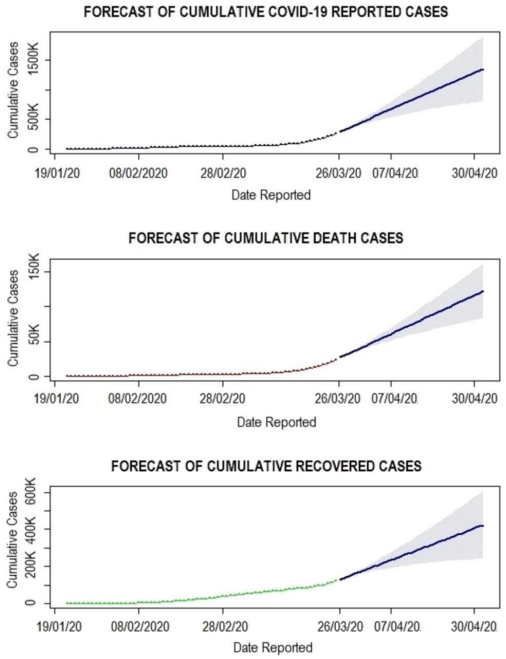

Figure 8: **Confidence intervals for COVID-19 indicators with ARIMA** [28]

An important detail that has not been mentioned so far: ARIMA and SARIMA models offer confidence intervals for future records, and therefore unknown values. This is shown in Fig. 8[19].

Now that the main differences between the two approaches have been explained and why they are still a widely used solution in industry and academia [20]. The following sections will illustrate the reader with more complex or novel solutions within the field of ML.

2.2 Decision Trees and Random Forest

Random Forest (RF) is a machine learning method based on an ensemble of decision trees that can be used for classification or regression. [29] [22]. It has been used succesfully in time series prediction, obtaining better results compared to the previously mentioned methods in cases where sufficient data were available. RF is a flexible and accessible machine learning algorithm that produces, even with

[19]The calculation of confidence intervals in these workspaces goes far beyond the points of interest of this Master's thesis, for more detailed information a very detailed resource is published by [4]

[20]Note that if they appear in this state of the art it is for that very reason, they are still a very widespread solution.

no hyperparameter optimisation, good results most of the time. In addition to the two examples listed above, there are many other examples in academia. Before explaining how this algorithm works, noted for its simplicity and diversity, there is another related concept that needs to be mentioned briefly: decision trees.

2.2.1 Decision trees

As the reader will already know, decision trees are a set of bifurcations, determined by the variables associated with the labels of the observations. Each decision tree will have a set of terminal nodes that will be the predictions or classifications [21] suggested by the trained tree. However, tree-based methods are not typically competitive with more complex approaches which will be mentioned later [21].

The objective of decision trees [22] is to reduce the error when segmenting the observations into j number of regions. Generally speaking, there are two steps to follow when constructing a Regression Tree (RT) [21]:

1. We divide the predictor space - that is, the set of possible values for $X_1, X_2, \ldots, X_p$ -into J distinct and non-overlapping regions, $R_1, R_2, \ldots, R_J$
2. For every observation that falls into the region R_j, we make the same prediction, which is simply the mean of the response values for the training observations in R_j. Theoretically, the distribution and complexity of the zones could be performed in any way.

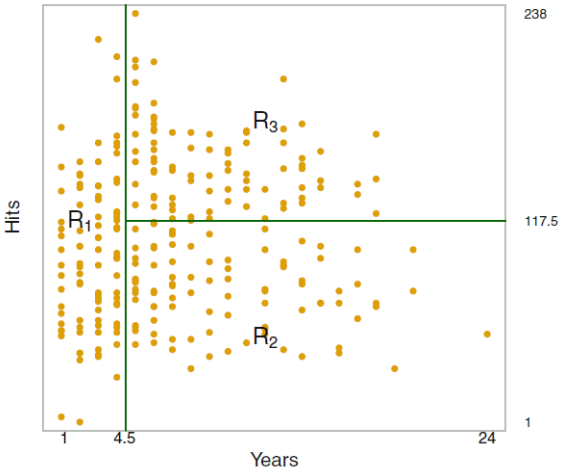

Figure 9: **An example of dividing observations according to their attributes with the objective of creating a decision tree** [21]

[21] Decision trees can be applied to regression and classification problems [21].
[22] Only the regression case will be instructed since we are dealing with time series

The main idea is to find the combination of boxes $R_1, ..., R_j$ that minimizes the following metric called Residual Sum of Squares(SRR)[23]:

$$SRR = \sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j}\right)^2 \tag{8}$$

Obviously, to consider every possible partition is computationally not feasible, specially working with several features and with thousand of rows (observations), therefore, there exist some approaches to solve this problem [24] [21]. Also, to avoid over-fitting[25], the tree is pruned by adding an α parameter that penalises the complexity of the model, being $|T|$ the number of final nodes in the tree[26]:

$$SRR = \sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T| \tag{9}$$

Now that the concept of the decision tree has been explained, it is possible to go a step further and talk about the random forest.

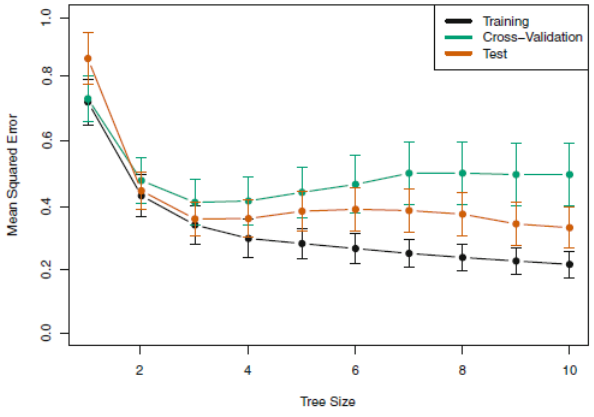

Figure 10: **The three with less error registered is the one with 3 finale nodes [21]**

[23] In the formula, $\hat{y_R}$ is that mean response for the training observation within the jth box.

[24] Explain this is out of the scope of this paper [10], this phenomena is shown in Fig. 10.

[25] Over-fitting is what happens when our model fits the noise of our training data by memorizing its peculiarities

[26] There are many concepts that are being overlooked in this explanation, and if the reader is unfamiliar with them, it is recommended the following resource [21].

2.2.2 Random Forest

By aggregating several decision trees and using Ensemble Machine Learning (EML) methods, researchers can improve the predictive power of their trees. One of them is called Random Forest (RF). EML consist on combining the predictions of several base estimators in order to improve generalizability and robustness over a single estimator [14], Hence its name as a "forest" made up of various trees.

Random Forest differentiates itself from other EML techniques by a little trick that decorrelates the generated trees from each other: If a variable accumulates all the importance in the set of regressors, most trees will be very similar, and therefore, no benefit will be found behind this computational cost [21]. Therefore, for each iterated model, RF will perform Boostrap as a sampling method. Bootstrapping is a sampling technique in which we randomly sample with replacement[27] from the data set [13]. Then, RF uses a limited amount of regressors for each Tree. The standard amount of this m number of these random features used for each tree is $p/3$ in regression problems[28] [42], where p is the the total number of attributes (columns) present in the data, for that reason it is included "random" in its name. This approach introduced by Leo Breiman [3] has several advantages, here we can list the main ones [25]:

- ✓ Low Bias, Moderate Variance
- ✓ It works well with both categorical and continuous values
- ✓ It automates missing values present in the data
- ✓ Normalising of data is not required as it uses a rule-based approach
- ✓ Quick Prediction/Training Speed

However, the following shortcomings need to be taken into consideration [25]:

- ✗ No model Interpretability[29]
- ✗ The saved tree can take up a lot of memory in some cases
- ✗ It can tend to overfit, if a cross validation is not performed

With regard to the last mentioned drawback, there is one last element to be mentioned from the RFs: they have a large number of parameters to optimise; once this task is done, very satisfactory results are obtained in time series [42].

[27] Sampling with replacement occurs when an observation is drawn from a finite population, and then, is returned to the whole data set. Thus, it is possible for a unit to be repeated in the new set of observations used to train the model.

[28] And $\sqrt{p}$ in classification.

[29] In the case of this project, this will not be a problem as we seek to make predictions that are as accurate as possible to reality, but in other cases this factor could be a determining factor in the choice of a model.

Hyperparameter	Description	Typical default values
mtry	Number of drawn candidate variables in each split	$\sqrt{p}$, $p/3$ for regression
Sample size	Number of observations that are drawn for each tree	n
Replacement	Draw observations with or without replacement	TRUE (with replacement)
Node size	Minimum number of observations in a terminal node	1 for classification, 5 for regression
Number of trees	Number of trees in the forest	500, 1,000
Splitting rule	Splitting criteria in the nodes	Gini impurity, *p* value, random

Figure 11: **Main elements to take into account when optimising parameters in the use of the RF algorithm [42].**

As the reader can appreciate, the algorithms shown in this scientific review are distributed from less to more complex. We see ML approaches that go far beyond human intuition, becoming the so-called Black Boxes(BB) of AI [25] [47].

2.3 Gaussian Process Regression

Gaussian Process Regression (GPR) is, roughly speaking, a non-parametric Bayesian machine learning technique that provides a flexible prior distribution over functions, enjoys analytical tractability, and has a fully probabilistic workflow that returns robust posterior variance estimations, thus, it makes possible to quantify uncertainty [46]. Unlike many of most recent state of the art machine learning frameworks, relies on few parameters to make predictions. Because GPR is (almost) non-parametric, it can be applied successfully to solve a wide variety of supervised learning problems, even when little data is available [50].

One conceptualization of a Gaussian Process (GP) is that it defines a distribution over functions [43]: given a GP, which is completely specified by a mean and covariance function, it is possible to sample a function at the point x from that Gaussian procedure [50]:

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}), k\left(\mathbf{x}, \mathbf{x}'\right)\right) \tag{10}$$

Here f(x) is the function obtained from the GP, m(x) is the mean function, and k(x, x') is a covariance function, which is equivalent to a kernel function in kernel methods[30] [50]. This is a formalization of sampling a random variable that depends on location x (in time series applications f(x) depends on time, t).

In order to turn Gaussian processes into powerful and practical tools it is necessary to develop procedures that address model selection problems [46]. In these mentioned methods there are included the discrete choice of the functional form for the covariance function [31]as well as values for two hyperparameters variance (σ) and step length (l) [32]. This is very well explained in a schematic and general way in Fig. 12, it illustrates the structure of a standard GP algorithm for use in energy quantification, thus, the objective is to predict future values of energy consumption in W (Power). This is done by specifying a prior (based on a carefully chosen kernel) and then refining the hyper-parameters iteratively based on a training sample, with the aim of generate a posterior. If the kernel is chosen correctly, predictions (with well-defined uncertainty) can be made about the behaviour of the energy system [32].

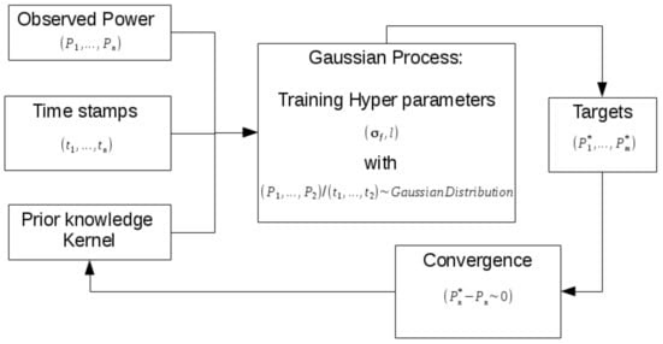

Figure 12: **description of how the time series prediction process works in GPR [32]**

Model selection

Another question arises now: which metrics will be used to perform the Kernel selection and the values of a set of hyperparameters θ? The objective is to optimize the **marginal likelihood** metric [46]. However, since this marginal likelihood has exponential terms, in order to derive an analytic gradient update [46] it will be chosen the parameter values which maximize marginal log-likelihood Eq. (11) and thus, marginal likelihood. After the GPR's hyperparameters have been optimize on the observed training dataset, the GPR model is ready to perfom posterior inference.

[30]Kernel Methods map our data into higher dimensional spaces with the aim of making our observations easier to separate or better to structure. [54].

[31]A multitute of possible families of covariance functions exist see section 4.2 in [46] for an overview.

$$\log p(\mathbf{y} \mid \mathbf{X}; \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^T \left[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2\right]^{-1} \mathbf{y} - \frac{1}{2}\log\left|K(\mathbf{X}, \mathbf{X}) + \sigma_n^2\right| - \frac{N}{2}\log 2\pi \qquad (11)$$

As mentioned above, GPR is a highly efficient technique with very little data available, and even without features of these records. This makes it a widely used technique in demand forecasting tasks. In Fig. 13, the good results in predicting energy consumption by Maritz (2018) are exposed, the figure illustrates a generic system with inter-daily load variations over a period of 5.1 days. The system suffered an energy intervention at t = 5, and the GP was used to predict the remained target data to compare the prediction with actual measured load [32], it can be seen that from t = 5.00 days GP allows a prediction of this demand with intervals that allow to understand the level of uncertainty of the prediction, according to the prior.

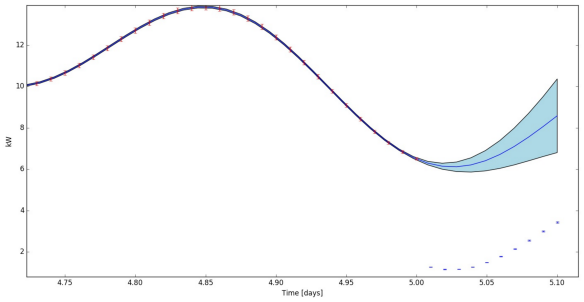

Figure 13: **Example of GPR for energy consumption forecasting** [32]

2.4 Recurrent Neural Networks and LSTM frameworks

Deep Learning (DL) is a sub-field inside AI and thus, inside Machine Learning [37]. Roughly speaking, DP is a form of ML that enables computers to learn from experience and understand the world in terms of a hierarchy of concepts [16]. DL works with artificial neural networks[32](NN) with the objective of achieving particularly efficient learning sequences.

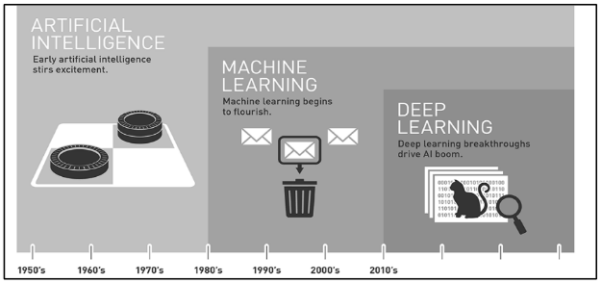

Figure 14: **Simple explanation of the evolution of AI over time** [9]

[32]Neural networks mimic the functioning of the human brain to obtain non-linear relationships in very complex problems (among other things) and have behind them a whole field of study, a highly recommended resource for a broad understanding of the concept is this book used in the context of a Master's degree in data science [8].

Like image classification or natural language processing, time series prediction can become a very difficult problem to solve in some situations [39]. The availability of workspaces with backpropagation[33] algorithms, and this in an open-source context, has allowed the implementation of this type of DL techniques, the customisation of network elements and the loss function [30] [1].

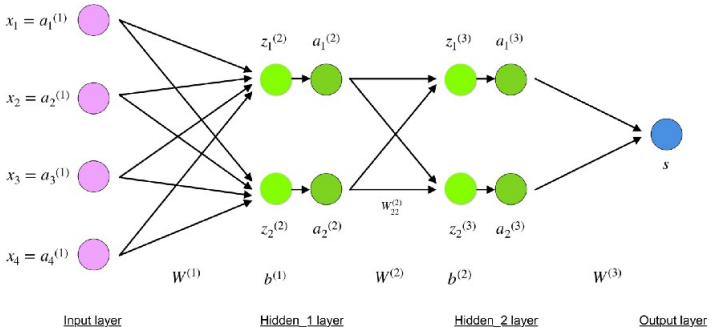

Figure 15: **Simple scheme of a 4-layer neural network** [26]

Given the diversity of time-series problems across several fields of knowledge, many NN frameworks have appeared [30]. A popular approach in academia is Recurrent Neural Networks (RNN), and its subtype Long Short-term Memory (LSTM) [30] [39].

2.4.1 Recurrent Neuronal Network

RNNs are basically NN with the ability to learn sequential problems, a time series would fall within these examples. The main idea behind RNN is using not only the input data, but also the previous outputs, with the aim of providing the network with a context [39]. So, the same input could produce a different output depending on previous inputs in the series.

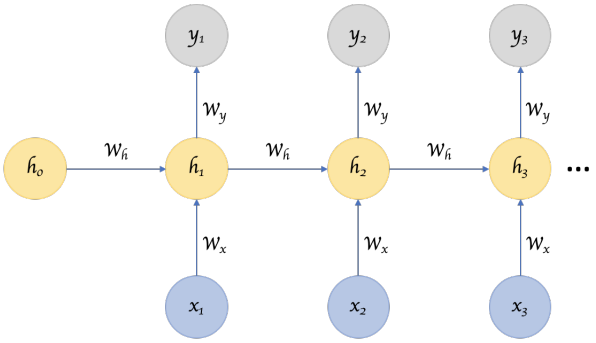

Figure 16: **A Recurrent Neural Network, with a hidden state that is meant to carry pertinent information from one input item in the series to others** [59]

[33] Back-propagation is a learning procedure for NN which adjust the weights of the connections in the network so as to minimize a measure of the difference between the actual output of the net and the desire output vector [48].

2.4.2 Long Short-Term Memory

LSTM is a 3-gated memory structure for NN, these three mentioned doors manage the content of the memory: they are simple logistic functions of weighted sums where the weights will be updated by backpropagation [39]. Therefore, LSTM fits perfectly into a NN scheme and its training mechanisms. LSTM can adapt it weights in these three ways, it can learn what to learn, what to remember and what to recall, without any special training or optimization [39].

3 Methodology

The motivations behind this research work and the main algorithms for time series prediction have been explained. Now it is time to gain a better understanding of the problem, for this reason different solutions have been used on the described ML problem, in a business context.

On the one hand, it is an interest key point of this project whether the different approaches proposed in the State of Art can be integrated into a Decision-Support System (DSS)[34] within a B2B company such as TS. On the other hand, it is also important to know which model works best, and to interpret why this is the case. For both questions valid answers have been found in the methods used. Figure 17 shows a schematic description of the methodology used in the ML experiments that have been carried out so far for this project.

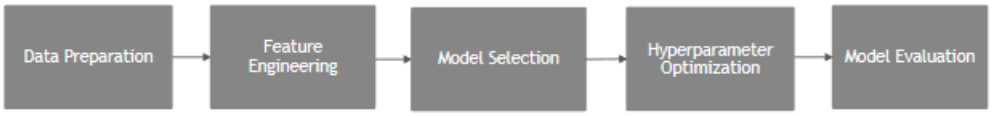

Figure 17: **The different stages of the Machine Learning process for selecting and tuning classical ML models** [45]

3.1 Data collection

TS has provided the data used in the methodology of this Master's Thesis. More specifically, historical data records of the cantabrian company . These tabular data is internal TS information, collected from its data warehouse through queries, so this is secondary data not collected during this research.

TipoDoc	Documento	LineaDoc	Factura	LineaFra
F	657453	1	657453	1
F	657454	1	657454	1
F	657455	1	657455	1
F	657455	2	657455	2

Table 1: **First five columns of TS provided data.**

Another important point to mention is the large amount of exploitable information in the data provided by this leader textile company. Unfortunately, due to the need to specify a first line of action, only a few columns of all the information available have been used.

[34]DSS represent a concept of computers within a decision-making process for any kind of organization. A DSS sifts through and analyzes massive amounts of data, compiling comprehensive information that can be used by organisation managers. [23]

Fecha	Cliente	PaisCliente	PaisDestino	Departamento
2001-01-24 00:00:00.000	999544	SP	PL	2
2001-01-24 00:00:00.000	999663	SP	PL	2
2001-01-24 00:00:00.000	999984	SP	PL	3
2020-01-24 00:00:00.000	999984	SP	PL	3

Table 2: **Columns 6 to 11 of TS provided data.**

The data observed in Table 3 now are records of all sales (F-value in the first column) and refunds (A-value in the second column) disaggregated by exact point in time, country of customer, country of destination and a number of other characteristics such as whether defects were found in the manufacture of the product. Below is an explanation of the types of data contained in each column used in the experiments mentioned in this research work: [35].

TipoOperacion	AfectaCantidad	Cantidad	Unidad	Precio
Venta	1	5.0	Metros	0.0
Venta	1	5.0	Metros	0.0
Venta	1	1.0	Metros	0.0
Venta	1	1.0	Metros	0.0

Table 3: **Columns 12 to 17 of TS provided data.**

- TipoDOC (First column), categorical variable: the value is 'F' for commercial invoices (money goes in) and 'A' values for refunds (money goes out), both of them could have multiple lines as related to the same single operation.
- Fecha (date of the commercial invoice), date variable[36]
- Importe (total value), float: Price in euro after using the conversion factor. Very easy to note that Quantity (Cantidad) * Price (Precio) * ExchangeRate (Cambio) = Total amount (Importe).
- GrupoArticulo (article group), categorical value: in this variable we will find discrete numeric values (1, 2, 3, 4,...) according to the category of the product sold (or returned in the register). This variable is not excessively important for the following sections as we are only going to focus on the records of family 2 for reasons that will be explained later on.

In addition to these columns of interest, in this tabular data, there are 33

[35]For a description of the rest of the variables, a brief description is provided in the GIT repository of this project, there are a total of 33 of them which can powered with ML in different ways, but as explained before, first lines of action in this TS project had to be delimited

[36]Note that a date in Python is not a data type of its own, but we can import modules like Pandas and/or Datetime to work with dates as date objects. [34].

columns and some of them make direct reference to product characteristics in that row, such as width (Ancho), number of defective metres, textile finishing (Acabado), print (Estampado), and more. It is important to note that this information could be used in further steps of this TS ML project.

GrupoArticulo	Serie	Acabado	Estampado	Ancho
3	4421	2568	0	145
2	40264	2783	0	145
4	162835	0	0	160
4	162835	0	0	160

Table 4: **Columns with GrupoArticulo column and other characteristics of the product.**

In the following section, the processes and tools used for the adequate preparation of the data will be explained, as well as the criteria for establishing the periods for which to aggregate the tabular data in order to generate the time series.

3.2 Data processing

To process tabular volumes of information such as those presented in the previous section, it is necessary to use tools suitable for these purposes, ranging from simple ones such as Microsoft Excel to more complex solutions prepared for large volumes of data [24]. In this particular case, Python [58] programming language has been chosen [37]. This software has libraries that are very suitable for carrying out this task performed in this Master's Thesis in a efficient way, from the computational and from the process automation point of view [24].

3.2.1 Popular Python Libraries used in the data processing

With this coding language, which is leading the field of Data Science [38][45], it is possible to use various libraries of interest used in the processing of data in this methodology. These libraries or packages, which are going to be be described below, have been used to generate a Python module called *info extraction*[39] which is used to extract aggregated sales volume data from the TS tabular data.

- Numpy 1.19.2[40]: provides high performance numerical and scientific computations on multidimensional arrays. It consists of many library functions and operations for the purpose of numerical computation. It maintains its good

[37] Other tools such as R programming language could be a good option, too.

[38] According to a recent KDnuggets poll thatsurveyed more than 1800 participants for preferences in analytics, data science, and, machine learning, Python maintained its position at the top of the most widely used language in 2019 [40].

[39] The code used for developing this module it is available in the project's GIT repository.

[40] Deriving its name from two words - Numerical Python [36].

performance even while providing a high-level abstraction for numerical computation [36] [17].

- Pandas 1.1.4: developers use it for loading data, preparing data, manipulating it, followed by its modeling, and then to analyze it. It is preferred among users because one is able to perform many time-consuming data science tasks such as Boolean indexing, checking for NaN's in a dataset, selecting and dropping a column from a dataset etc. All these tasks does not require loops [36] [34].
- Math 1.0.1: Python math module provides access to the mathematical functions defined by the C standard [58].
- Glob 7.1.7: the glob module find all the path-names matching a specified pattern according to the rules used by the Unix shell. This module is very important when we need to extract tabular aggregated information from more than one file.

Info extraction developed for this and similar future projects of TS allows the user to extract aggregated data about the total sales or refunds from any kind of product family (or all of them). Also, this module created for TS also allows the company to choose between generating a time series of total sales (or returns) on a weekly or monthly basis, as will be seen later, depending on the model implemented, it will be of interest to choose one frequency of registration or another. For instance, a simple and generalist run that could be requested from the methods within the module would be a monthly time series of the total quantities invoiced for products of all classes manufactured by TS.

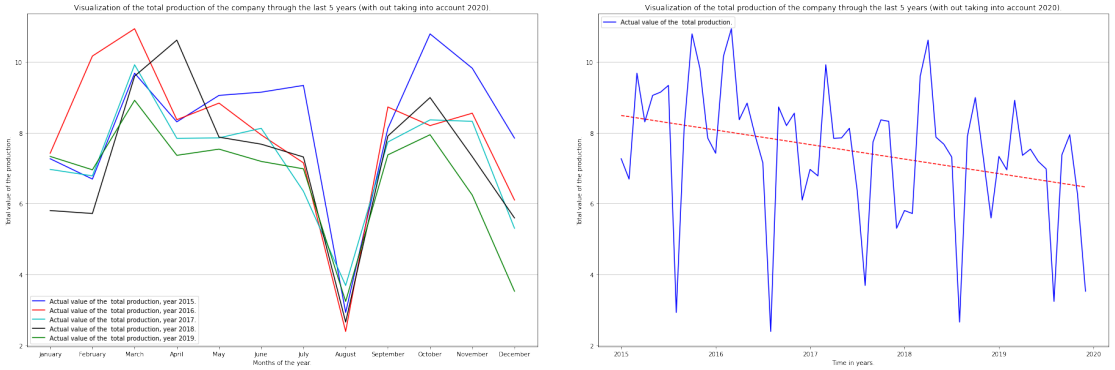

Figure 18: **Visual comparison between years of the total value of the production of TS from 2015 to 2019.**

3.2.2 Time series data samples

After all, The objective of the current project is to investigate if ML can be used to assist production decisions in TS factories. In Fig. 18 it is possible to see how the total production shows a trend and certain peaks in total demand which occur at specific times of the year. This is very useful for drawing general conclusions, however, in order to be able to make specific decisions it is needed to disaggregate by product families, and ideally even by product. In other words, knowing with a

more or less acceptable accuracy the quantity demanded within a business line at a given time allows to buy the optimal amount of raw materials from suppliers and to optimise the use of the warehouse, this kind of decisions can save a company in such a competitive environment as the one TS operates in. Therefore, for our ML experiments, two time series were generated through the *info extraction* module as follows:

1. The total turnover has been calculated for each time unit of the time series only for products of family two as this is the business line that shows a more cyclical behaviour and, therefore, it is of great interest for TS to know its fluctuations throughout the year with precision.
2. Given that TS production stress has been atypical during 2020 due to the COVID-19 pandemic. The query obtained records from January 2015 to December 2019.
3. The ML experiments were conducted with two time series, one weekly and one monthly. In the weekly case, there is more variability in the data, which makes it more difficult to predict, but this increased the sample size.

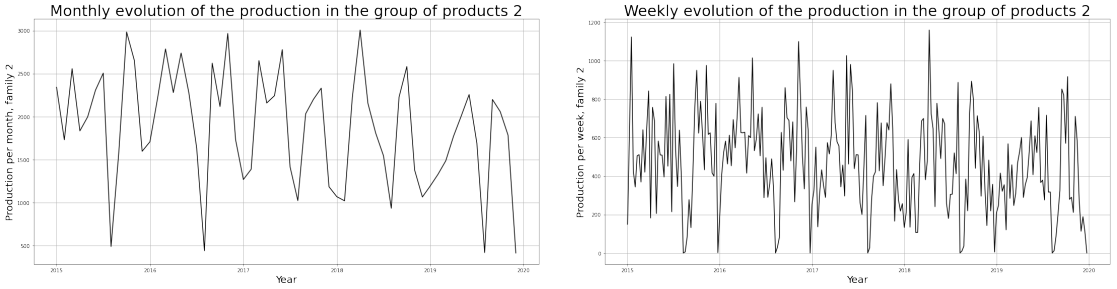

Figure 19: **Comparison between the business line 2 time series on weekly and on a monthly basis.**

3.3 Feature engineering

The predictability of the future value of a quantity depends on several components, including [19]:

1. How well we understand the factors related to it.
2. How much data is available.
3. Whether the forecast can affect the thing we are trying to forecast.

In this case, the second [41]and third components are not a problem, however, it is difficult to know the future value of indicators that may affect textile family

[41]It is true that in the case of the monthly time series there are 60 (12 multiplied by 5) records and in the weekly time series there are 240 (48 multiplied by 5), which is too few data for Deep Learning models, but as will be shown below, for other solutions this is not a problem.

2 demand such as GDP growth next year or whether a global pandemic will occur next month. Smart feature engineering can help.

Therefore, when long-run forecasts are required, there are very few external and internal indicators whose value we will know with certainty. Time is one such thing. It is possible use it as a regressor for our model for multiple steps ahead [39], thus, it is possible to use the the seasonal component of the time series as a very powerful tool. Therefore, the mentioned *info extraction* module generates at the same time as the time series data, its corresponding features, in this case the module has been designed to generate associated to each time unit $y_t$the following values to be used as regressors, this data has been used a predictors in the ML performed during this project:

3.3.1 Lags

Lagging basically means going steps back in time. The past could be a very good resource to predict the future values of our target variable y, that is why ARIMA or SARIMA frameworks have an auto-regressive component. The only inconvenient of implementing this as a regressor is that a p order of values is going to be lost [39]. In this case, lags of first order have been used [42].

3.3.2 Trend

Trend indicates the passage of time, a sigle variable equidistant increasing numbers serves that purpose [39]. This has been considered as well in our models as a predictor.

3.3.3 Seasonality

In business problems we usually find repeated patterns from year to year, as we have seen in the case of SARIMA or with HWSM, the moment of the business year in which we find ourselves in certain occasions can be a very powerful regressor. Just like the trend, it is a value that will be known in the future, and then, it could be very reliable in long run predictions.

One approach to capturing seasonality as a regressor in econometrics is to use a dummy variable for each month (minus one of them), however, to use the values of the months (from one to 12) within a continuous scale has be chosen in this case. In order not to have too far away the value on this scale from the first month to the last one, two variables related to seasonality were created: one using a sine transformation Eq. (12) and the other a cosine transformation Eq. (13) [39].

[42] Another interesting idea to be able to make predictions more in the long term would be to use lags of order 13 in the monthly case and 48 in the weekly case, however, with the current size of our time series it was considered sufficient to use as predictors the values recorded in $y_t - 1$.

$$\tilde{x} = \sin\left(\frac{2 \cdot \pi \cdot x}{\max(x)}\right) \tag{12}$$

$$\tilde{x} = \cos\left(\frac{2 \cdot \pi \cdot x}{\max(x)}\right) \tag{13}$$

It is very convenient to add the seasonal component to the model as the decomposition of both time series used shows it, in addition to a trend due to the passage of time. This decomposition is in Fig. 20. Note again the trade off between a time series with more residuals (fewer factors that we can explain) and one with fewer observations.

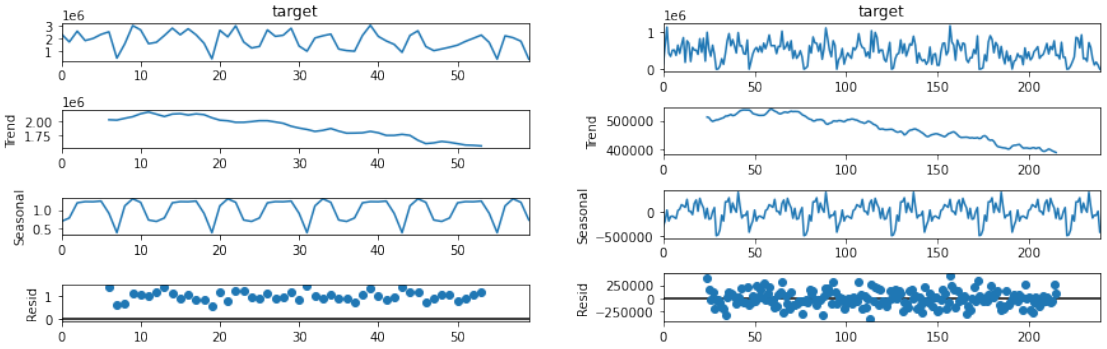

Figure 20: **Comparison of the descompositions of the time series on a monthly basis (left) and of the time series on weekly basis (right).**

Feature and target standardization

Features are usually normalized before being used to train a NN. It is beneficial for the training process [39]. The standard scaler Eq. (14) has been chosen for the target variable y (total value of demand for the business line 2) and the set of predictors X for each time unit y_t of these ML experiments. Feature Scaling is very important for algorithms that use gradient descent like linear regression, logistic regression, neural network..etc. and also for algorithms that use Euclidean distance like K-Nearest Neighbors[43] [41].

$$\tilde{x} = \frac{x - \text{mean}(x)}{sd(x)} \tag{14}$$

[43]However, Tree Based approaches do not require feature scaling.

3.4 Model implementation

Based on the state of the art, different models were trained with the aim of validating them later, since the interesting thing about a machine learning model is not that it has very good results on its training sample, but that it can be used to generalise and predict new observations with unknown labels. This final section of the methodology will explain how and based on what criteria decisions were made regarding the implementation of models: from how to split our sample to reserve some observations for validation, to which metrics to choose to know which approach is working best and so on.

3.4.1 Sample partition:train and test

A random partitioning of the sample into train and test has been standard procedure within the ML expert community for decades [56]. In this case, the partitioning of the 5-year sample was performed out as follows:

1. **Train split**: the first four years available, in the case of the time series by weeks being 192 observations followed against 60 for the time series by months, i.e. we will train the models, with the cumulative sales data by months/weeks from January 2015 to December 2018 (80% of our data).
2. **Test split**: the year 2019 was reserved for examining the predictive ability of the trained models, thus about 48 observations in the case of the weekly time series, and 12 in the monthly case (20% of the data).

As discussed above, it is considered important to train and examine our models with non-outlier years, so a time series involving 2020 has not been used. The 20/80 partition is one of the options commonly repeated in ML experiments by experts [7]. In this case, we have chosen this approach, as we wanted to test how effective the state-of-the-art models could be in acting on an entire TS business year, from January to December, our last non-atypical year was therefore 2019.

3.4.2 Base line predictors.

In order to better understand the present problem, two naive predictors (or base line predictors) were included in the developed module *info extraction*. This process in the methodology helped to test whether the state of the art models subsequently employed perform better than the more simplistic reasoning; this helped to determine whether it was convenient to implement ML in the future to solve TS business problems. These two naive predictors used are based on:

- **predict_with_last_year**: this function inside the project's Python module just takes the last year of the training data as a prediction. In other words, this naive predictor establishes that the value of the total production in January in 2018 is going to be the same in January 2019, the same February and so on.

This is something that a human being could quickly reason out in a simple way.

- **predict_with_ema**: this other Module's function is not as simple as the last one. For any time unit in the test year (2019) it works out the Exponential Moving Average (EMA) of the same time units in the previous years (training data) after optimizing an α parameter —for example, the EMA of all the February values in our last years (2015-2018)—. This means, this predictor at least knows that if something happened in August last year, it does not have to be the same in August 2019, unless, it occurs every August in all our time series[44].

$$EMA_t = \begin{cases} x_0 & t = 0 \\ \alpha x_t + (1-\alpha) EMA_{t-m} & t > 0 \end{cases} \tag{15}$$

Both approaches have been implemented with the aim described above. In the following section we will discuss the models implemented and the software used to run them.

3.4.3 Python libraries for model implementation

For ML model training and testing process, the Python programming language has continued to be used, in this case now in Jupyter Notebook. Thanks to its widespread use and community, Python has a wide variety of libraries related to statistical learning and ML. In this ML project, the following packages were used [45]:

- **Statsmodels, 0.12.2**: is a Python module that provides classes and functions for the estimation of many different statistical models [51]. It has implemented classes for SARIMA and seasonal HW, very convenient for this specific case, due to its proven seasonality.
- **Scikit-learn, 0.23.2**: scikit-learn is an open source machine learning library written in Python. It allows the easy and fast integration of machine learning methods into Python code [27]. Thanks to this library it was possible to quickly and efficiently implement data mining models such as multiple regression, KNN, and even Random Forest in the experiments [38].
- **TensorFlow, 2.3.0**: is an open source library used primarily for deep machinelearning. It was originally developed on by Google's divisions, but in 2015 itwas released as free open source software [15]. As the reader can imagine, this framework was used for the application of LSTM NN [1].

[44] Note again in Eq. (15) that m means the total units in which the year is divided (months 12 and weeks 48), so, the EMA was worked out from the same weeks or months in the last years.

[45] All of them are widely spread in the ML expert community, see Fig. 21.

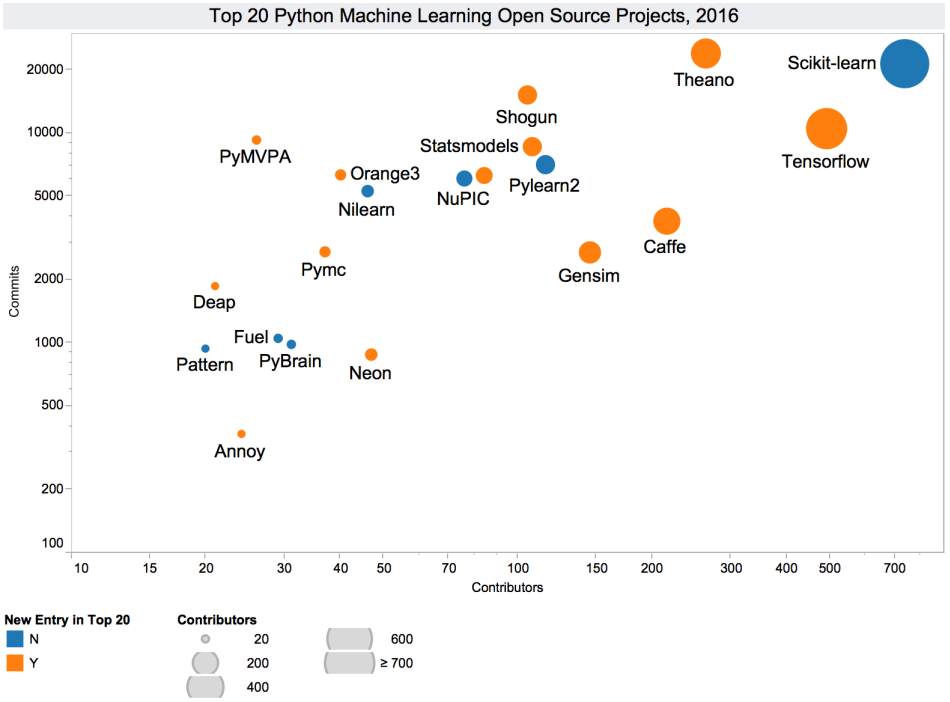

Figure 21: **Most used Python libraries in 2016** [6].

3.4.4 Evaluation metrics: error and goodness of fit

Another factor to take into account is which measures have been followed in order to interpret the results. In this case, two metrics have been used according to different criteria:

- **Root Mean Square Error** (RMSE): is the error metric to minimize, this approach has been chosen over Mean Square Error (MSE), because, RMSE is better in terms of reflecting performance when dealing with large error value and he RMSE is directly interpretable in terms of measurement units.

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{16}$$

- **Coefficient of determination** (R^2): measures the proportion of variance explained by the model. This correlation, known as the "goodness of fit," is represented as a value between 0.0 and 1.0.

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{17}$$

Another necessary point is to answer the following question: how can someone compare the RMSE between a model applied to the weekly time series and a different one applied to the monthly time series — without any previous

modification in the error of monthly model this doesn't make sense —? The approach chosen was to divide the error of the models when trained with monthly models by 4, so that we would always be talking about the same unit: thousands of euros per week.

3.4.5 Hyperparameter tuning

In the case of certain algorithms, some of their hyperparameters were optimised with cross-validation within the train sample using the mentioned software Scikit-learn and its methods *cross_val_score* and *GridSearchCV* [38]. These concrete hyper-parameter optimisations were performed on three models, two from Scikit-learn and one from Tensorflow: number of neurons in the intermediate layer in the LSTM model, the wide range of random forest hyperparameters described in the state of the art and the number of near neighbours in KNN[46].

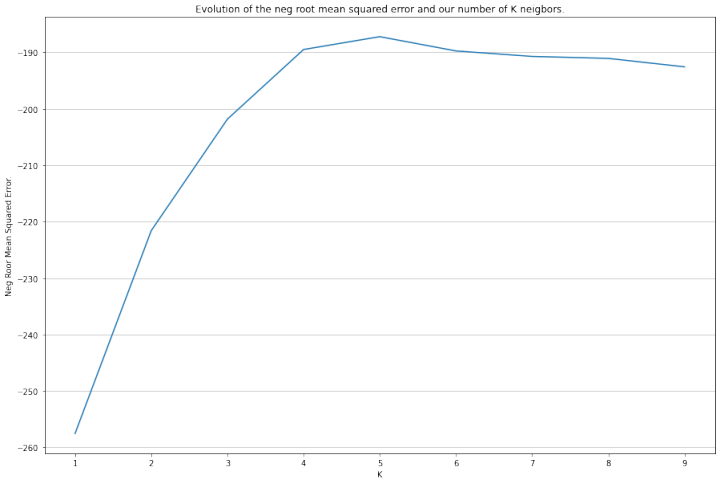

Figure 22: **Optimization of the k number of neighbours to consider in the KNN model trained with our weekly time series, being the optimum number to consider at the 5 closest observations.**

[46] Although this approach is not mentioned in the State of the Art, this framework can also obtain very good results with time series [33]

4 Results

In this section, the results of the models implemented in the methodology are shown in the following table. Subsequently, in the discussion section, special emphasis will be placed on the possible causes and implications of the metrics obtained for future ML projects in the TS company.

Firstly, it is convenient to show the time series on which the mentioned ML models have been trained. This time series has been created as a result of the *info extraction* module. After all, this is a result of the methodology used as well[47], see in Fig. 23, how the weekly time series looks.

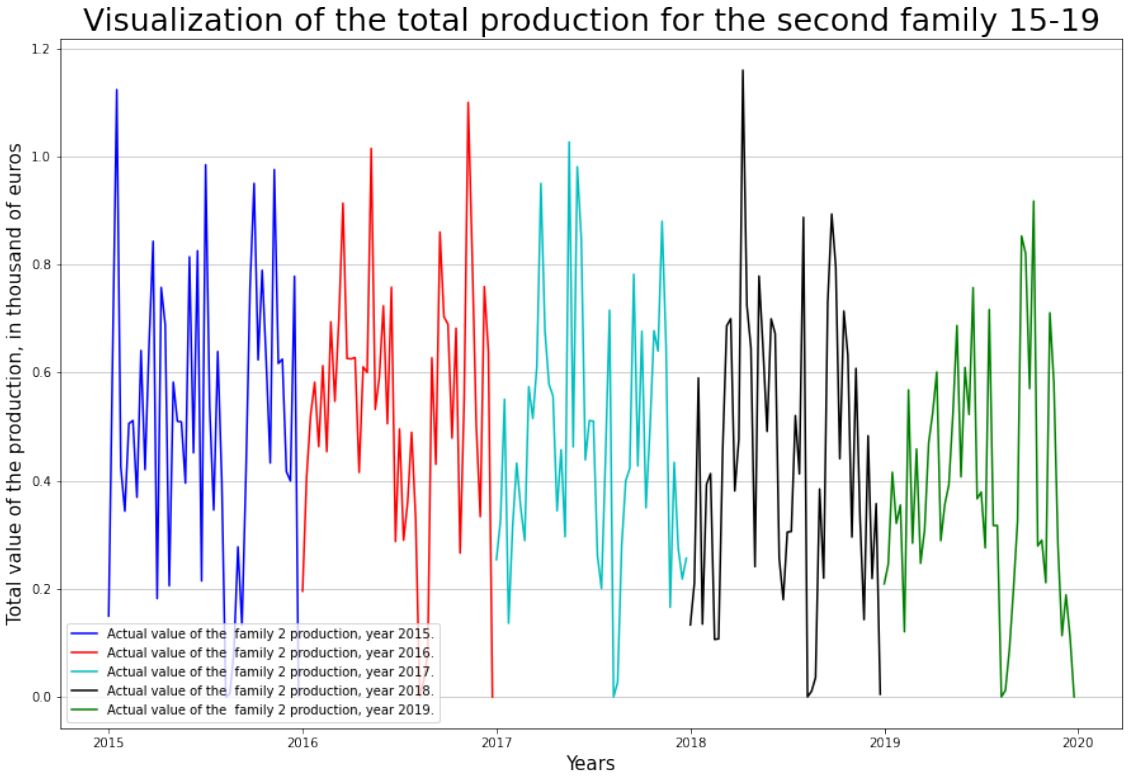

Figure 23: **Weekly time series generated by the info extraction module from the TS data.**

Second, the implementation of the base line predictors has served as a guide to to understand the complexity of the problem at hand. On the one hand, the naive predictor *predict_with_last_year* - which simply infers that the observations in a year to be predicted will be the same as in the previous year - made possible to see the weekly similarity between one year and its consecutive year, in this case between 2018 — the last year of training — and 2019 — the test year — see Fig. 24.

[47]Note, the *info extraction* has been developed as part of the methodology during this TS project.

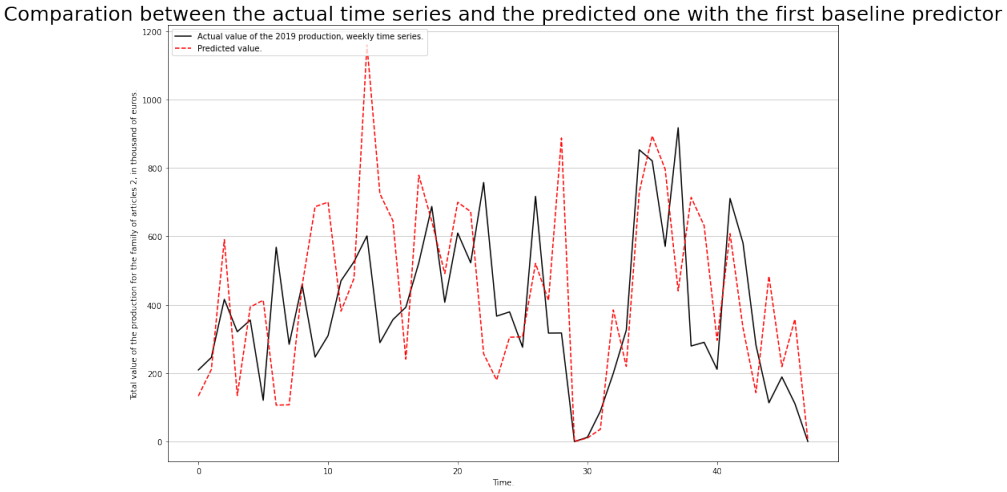

Figure 24: **Predicting with the last year.**

On the other hand, another naive prediction with the EMA of each week from previous with an optimised alpha parameter — see Fig.25 —, has shown the relationship at a glance between a business year and previous periods.

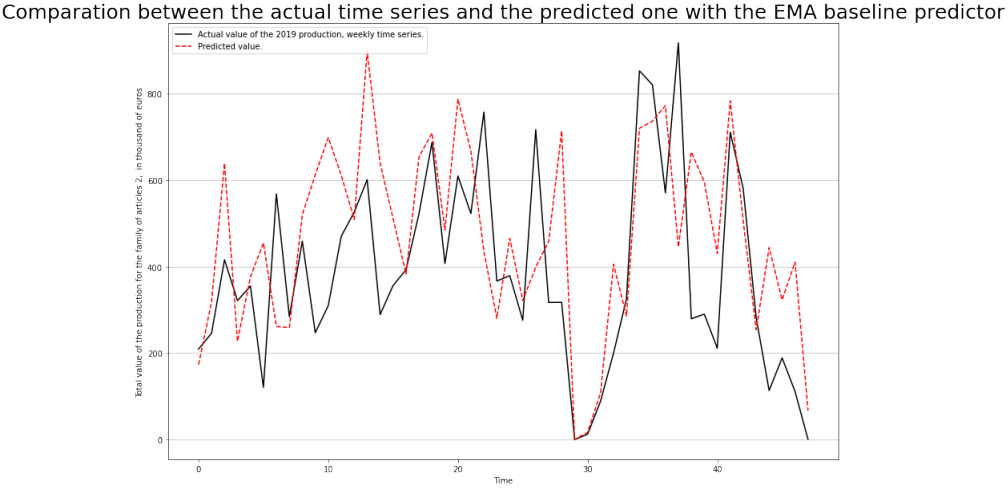

Figure 25: **Predicting with the same week's EMA from previous years.**

In order to better understand this problem, the models explained in the state of the art have been implemented with the frameworks mentioned in the methodology.

Finally, in the following tables it is possible to observe the values obtained for the validation metrics mentioned in the methodology: in Table 5 we can observe the RMSE error metric for each implemented model, and the goodness of fit metric of each value in both time series in Table 6. Note that the most naive models have been implemented only on the most specific time series - the weekly one - and no NN

has been trained with the monthly time series because it was pointless for reasons explained in the Discussion section.

Framework	Weekly time series	Monthly time series
Last year (naive)	253.64	-
EMA (naive)	211.45	-
HWSM	203.88	130.5
SARIMA	**52.82**	**98.56**
Linear regression	207.99	153.28
KNN	199.58	144.57
Random forest	171.07	139.05
GPR	292.53	108.85
LSTM	243.07	-

Table 5: **Mean error for each implemented model in thousand of Euros per week (RMSE).**

Framework	Weekly time series	Monthly time series
Last year (naive)	0	-
EMA (naive)	0.11	-
HWSM	0.17	0.23
SARIMA	**0.80**	**0.87**
Linear regression	0.14	0
KNN	0.08	0.05
Random forest	0.64	0.13
GPR	0	0.47
LSTM	0	-

Table 6: **Goodness of fit values for each implemented model (R_2).**

5 Discussion

The best insight that could be deduced from the results obtained is that a small fraction of a company's history — in this case 4 business years have been used to train the mentioned models — can be used to apply Machine Learning models with results that are better than simple intuition and with an accuracy that is as desirable as weekly. However, much more data is needed to train innovative and attractive solutions such as LSTM networks. In Fig. 26 below it is possible to observe how the neural network cannot infer non-linear relationships with so few points available, this is the case for the weekly time series, in the monthly case it would be even worse.

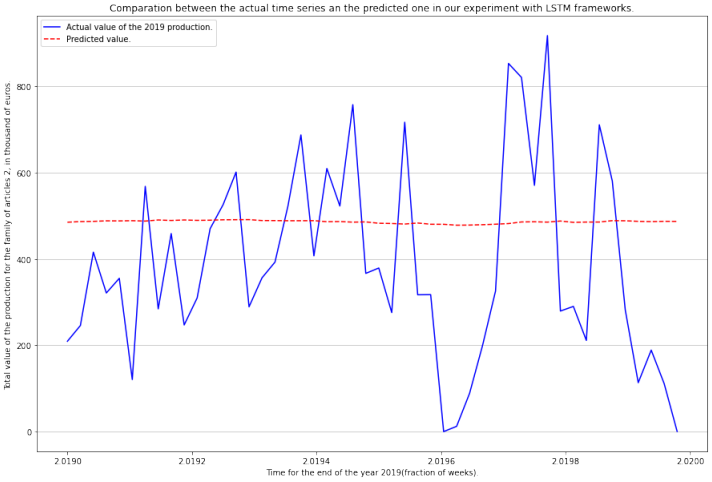

Figure 26: **LSTM framework when few data is available.**

Another fact demonstrated in the application of the methodology is the good performance of the HWSM and SARIMA models, very good choices due to the displayed components of the time series, thus obtaining the best error and explanatory metrics. Although its best results are with the monthly time series -Fig. 27, left-, as it has a lower residual, SARIMA also achieves a coefficient of determination of 0.8 with the weekly data - thus, a 80% of the variability of the model could be explain with the SARIMA model for weeks -, see Fig. 28, right.

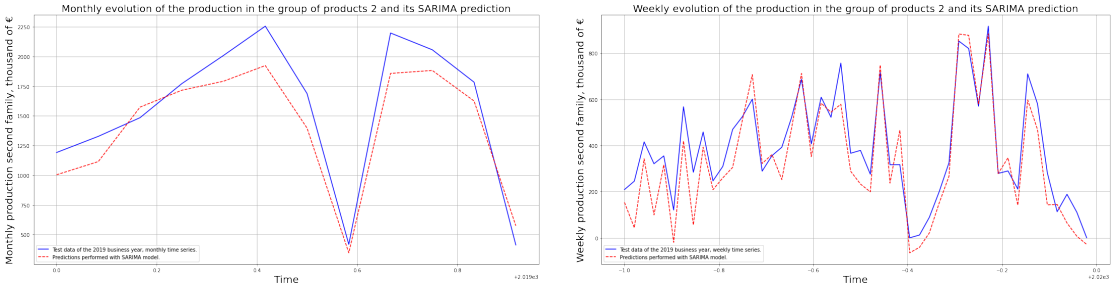

Figure 27: **Visual comparison between applying SARIMA to the the monthly time series (less residuals) and the data on a weekly basis (more observations).**

Another model with very good results has been RF, with an explanatory power of up to variability 64% of the data. The expert knowledge and the graphs, especially the monthly one (see Fig.29), suggest that the results could be improved by incorporating holidays in client countries [39].

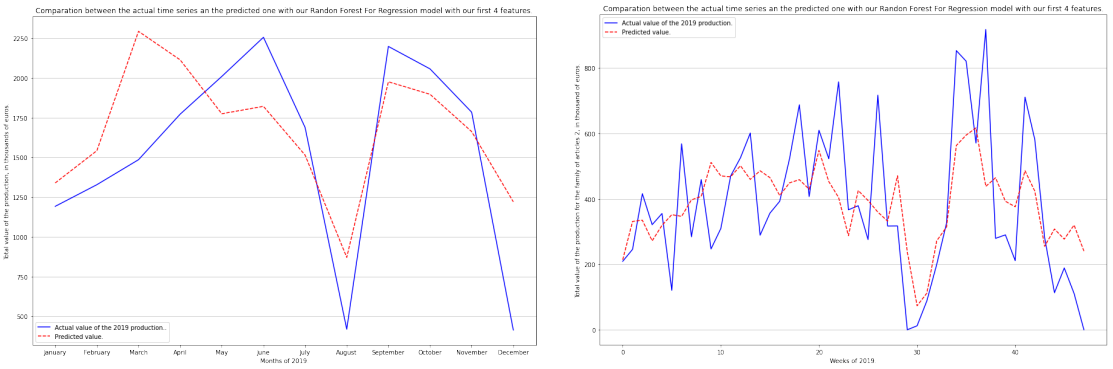

Figure 28: **RF predicting results on the monthly data (left), and on the monthly time series (right).**

Also, in the weekly case, linear regression and KNN have obtained less error and more explainability. However, as it is mentioned, RF will be the head of the pack when more data is available.

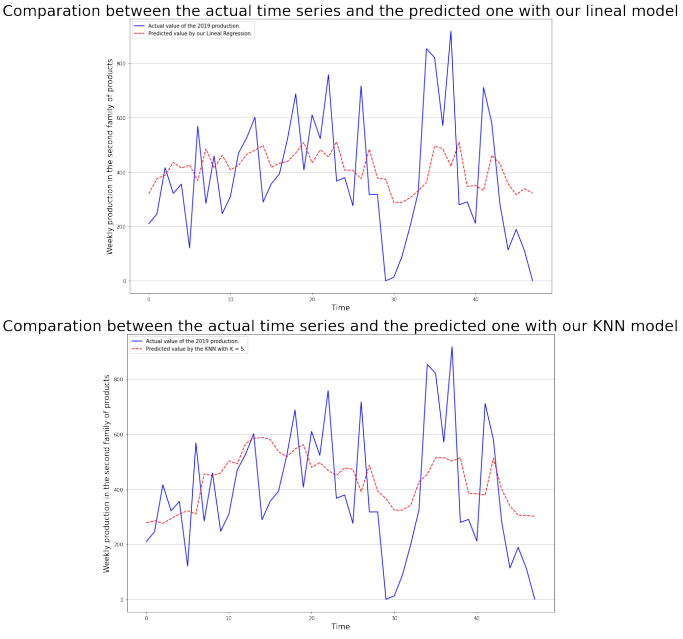

Figure 29: **Prediction of the 2019 business year by the linear (top) and KNN (down) data mining frameworks.**

Despite not having the best results, a very attractive and robust solution is GP, as it allows the quantification of uncertainty in confidence intervals[48]. After all, this is what allows decision making within companies, not predicting float values. In this case, the model trained with monthly data is preferred as it has less noise.

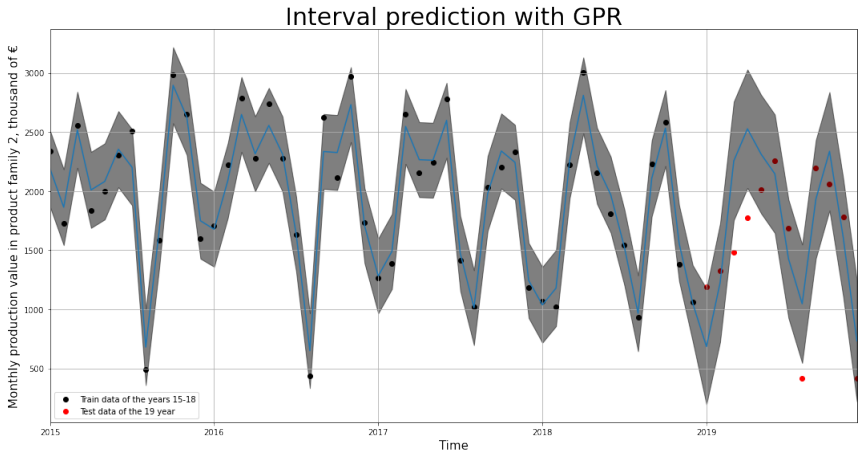

Figure 30: **GP applied to the monthly time series.**

For the better understanding of the problem at hand, the implemented methodology has shown that various ML approaches can work better than simple intuition, and therefore, such techniques can help in the search for efficiency within companies. However, more ambitious and innovative models such as RF or LSTM would require more data and more features, such as the aforementioned dummies with state holiday dates. Probably, with these changes even better results would be obtained. A framework to apply the suggested feature would be Prophet, from Facebook [49] [57].

[48] As mentioned in the state of the art, ARIMA models also have this advantage.

[49] This library is also being widely used nowadays due to its ease of implementation and good performance in time series problems.

6 Conclusions

This study has discussed the most widely used and promising models for predicting time series, and thus for predicting future demand values in a business context. Different components of these processes can be used for their future prediction, in addition to the autocorrelation of their own values or external factors.

In particular, a methodology was implemented with the aim of creating time series based on disaggregated company information. In order to have a base from which to measure the difficulty of the problem to be solved, very simple and naive predictors have been used, such as predicting a year by repeating the observations of the previous year. These naive predictors have been compared with Machine Learning models with different degrees of complexity and different data requirements.

On the one hand, the most statistically robust approaches such as SARIMA or HW Seasonal Model have obtained the best results both in terms of explaining the variability of the data (R_2) and reducing the average error of the predictions ($RMSE$).

On the other hand, other more complex approaches might give better results with larger amounts of data, such as LSTM recurrent networks. Furthermore, in order to improve results, it would be very interesting to create new features such as dummy variables for important dates that could affect future demand values. The implementation of hybrid models could even be considered [53].

Finally, the results obtained with RF are also very promising as they can replicate the variability of the data so well with weekly precision, and GP gives a very valuable asset in decision making such as the quantification of uncertainty.

Regarding next steps, the long-term goal is to automate the implementation of these models, and even the implementation of more approaches in order to be able to make just-in-time demand forecasts, or the elaboration of budgets for purchases from suppliers.

The implementation of the code to create the project module and the implementation of the experiments performed for this Machine Learning enterprise project can be reviewed in the GitHub repository[50] of this TS project.

[50] https://github.com/javi-saez/TFM-Textil-Santanderina

References

[1] Martín Abadi et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems". In: *arXiv preprint arXiv:1603.04467* (2016).

[2] Jason Bell. *Machine learning: hands-on for developers and technical professionals*. John Wiley & Sons, 2020.

[3] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[4] Peter J Brockwell et al. *Introduction to time series and forecasting*. Springer, 2016.

[5] Robert Goodell Brown. *Statistical forecasting for inventory control*. McGraw/Hill, 1959.

[6] Jason Brownlee. *Top 20 Python Machine Learning Open Source Projects, updated*. https://www.kdnuggets.com/2016/11/top-20-python-machine-learning-open-source-updated.html. [Online; accessed 07-July-2021]. 2016.

[7] Jason Brownlee. *Train-Test Split for Evaluating Machine Learning Algorithms*. https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/. [Online; accessed 07-July-2021]. 2020.

[8] Enrique Castillo et al. "Introduction to Neural Networks". In: *Functional Networks with Applications*. Springer, 1999, pp. 5–46.

[9] Michael Copeland. *What's the Difference Between Artificial Intelligence, Machine Learning and Deep Learning?* https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/. [Online; accessed 25-June-2021]. 2016.

[10] Tom Dietterich. "Overfitting and undercomputing in machine learning". In: *ACM computing surveys (CSUR)* 27.3 (1995), pp. 326–327.

[11] Naveen Donthu and Anders Gustafsson. "Effects of COVID-19 on business and research". In: *Journal of business research* 117 (2020), p. 284.

[12] Gonghao Duan and Ruiqing Niu. "Lake area analysis using exponential smoothing model and long time-series landsat images in Wuhan, China". In: *Sustainability* 10.1 (2018), p. 149.

[13] Bradley Efron. "Bootstrap methods: another look at the jackknife". In: *Breakthroughs in statistics*. Springer, 1992, pp. 569–593.

[14] Xianwei Gao et al. "An adaptive ensemble machine learning model for intrusion detection". In: *IEEE Access* 7 (2019), pp. 82512–82521.

[15] Migran N Gevorkyan et al. "Review and comparative analysis of machine learning libraries for machine learning". In: *Discrete and Continuous Models and Applied Computational Science* 27.4 (2019), pp. 305–315.

[16] Ian Goodfellow et al. *Deep learning*. Vol. 1. 2. MIT press Cambridge, 2016.

[17] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585 (2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2.

[18] Charles C Holt. "Forecasting seasonals and trends by exponentially weighted moving averages". In: *International journal of forecasting* 20.1 (2004), pp. 5–10.

[19] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[20] Garima Jain and Bhawna Mallick. "A study of time series models ARIMA and ETS". In: *Available at SSRN 2898968* (2017).

[21] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[22] Michael J Kane et al. "Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks". In: *BMC bioinformatics* 15.1 (2014), pp. 1–9.

[23] Peter GW Keen. "Decision support systems: a research perspective". In: *Decision support systems: Issues and challenges: Proceedings of an international task force meeting*. 1980, pp. 23–44.

[24] Birendra Khadka. "Data analysis theory and practice: Case: Python and Excel Tools". In: (2019).

[25] Julia Kho. *Why Random Forest is My Favorite Machine Learning Model*. https://towardsdatascience.com/why-random-forest-is-my-favorite-machine-learning-model-b97651fa3706. [Online; accessed 21-June-2021]. 2019.

[26] Simeon Kostadino. *Understanding Backpropagation Algorithm*. https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd. [Online; accessed 25-June-2021]. 2019.

[27] Oliver Kramer. "Scikit-learn". In: *Machine learning for evolution strategies*. Springer, 2016, pp. 45–53.

[28] Pavan Kumar et al. "Forecasting the dynamics of COVID-19 pandemic in top 15 countries in April 2020: ARIMA model with machine learning approach". In: *MedRxiv* (2020).

[29] Deukwoo Lee and Soowon Lee. "Hourly Prediction of Particulate Matter (PM2. 5) Concentration Using Time Series Data and Random Forest". In: *KIPS Transactions on Software and Data Engineering* 9.4 (2020), pp. 129–136.

[30] Bryan Lim and Stefan Zohren. "Time series forecasting with deep learning: A survey". In: *arXiv preprint arXiv:2004.13408* (2020).

[31] Pankaj Malhotra et al. "Long short term memory networks for anomaly detection in time series". In: *Proceedings*. Vol. 89. Presses universitaires de Louvain. 2015, pp. 89–94.

[32] Jacques Maritz, Foster Lubbe, and Louis Lagrange. "A practical guide to Gaussian process regression for energy measurement and verification within the Bayesian framework". In: *Energies* 11.4 (2018), p. 935.

[33] Francisco Martı´nez et al. "A methodology for applying k-nearest neighbor to time series forecasting". In: *Artificial Intelligence Review* 52.3 (2019).

[34] Wes McKinney et al. “Data structures for statistical computing in python”. In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX. 2010, pp. 51–56.

[35] Elizbar A Nadaraya. “On estimating regression”. In: *Theory of Probability & Its Applications* 9.1 (1964), pp. 141–142.

[36] Abhinav Nagpal and Goldie Gabrani. “Python for data analytics, scientific and technical applications”. In: *2019 Amity international conference on artificial intelligence (AICAI)*. IEEE. 2019, pp. 140–145.

[37] Daniel Paschek, Caius Tudor Luminosu, and Anca Draghici. “Automated business process management–in times of digital transformation using machine learning or artificial intelligence”. In: *Matec web of Conferences*. Vol. 121. EDP Sciences. 2017, p. 04007.

[38] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[39] Gábor Petneházi. “Recurrent neural networks for time series forecasting”. In: *arXiv preprint arXiv:1901.00069* (2019).

[40] Gregory Piatetsky. “Python leads the 11 top Data Science”. In: *Machine Learning Platforms: Trends and Analysis* 2019 (2019).

[41] Ashwinh Prasad. *Why do Feature Scaling ? — Overview of Standardization and Normalization — Machine Learning*. https://medium.com/analytics-vidhya/why-do-feature-scaling-overview-of-standardization-and-normalization-machine-learning-3e99d16eeca8. [Online; accessed 06-July-2021]. 2021.

[42] Philipp Probst, Marvin N Wright, and Anne-Laure Boulesteix. “Hyperparameters and tuning strategies for random forest”. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 9.3 (2019), e1301.

[43] Joaquin Quinonero-Candela and Carl Edward Rasmussen. “A unifying view of sparse approximate Gaussian process regression”. In: *The Journal of Machine Learning Research* 6 (2005), pp. 1939–1959.

[44] Jayanthi Ranjan. “Business intelligence: Concepts, components, techniques and benefits”. In: *Journal of theoretical and applied information technology* 9.1 (2009), pp. 60–70.

[45] Sebastian Raschka, Joshua Patterson, and Corey Nolet. “Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence”. In: *Information* 11.4 (2020), p. 193.

[46] Carl Edward Rasmussen. “Gaussian processes in machine learning”. In: *Summer school on machine learning*. Springer. 2003, pp. 63–71.

[47] Anderson Rocha, Joao Paulo Papa, and Luis AA Meira. “How far do we get using machine learning black-boxes?” In: *International Journal of Pattern Recognition and Artificial Intelligence* 26.02 (2012), p. 1261001.

[48] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), pp. 533–536.

[49] Stuart Russell and Peter Norvig. *Künstliche Intelligenz*. Vol. 2. Pearson Studium München, 2012.

[50] Ryan Sander. *Gaussian Process Regression From First Principles*. https://towardsdatascience.com/gaussian-process-regression-from-first-principles-833f4aa5f842. [Online; accessed 29-June-2021]. 2015.

[51] Skipper Seabold and Josef Perktold. "statsmodels: Econometric and statistical modeling with python". In: *9th Python in Science Conference*. 2010.

[52] Alex J Smola and Bernhard Schölkopf. "A tutorial on support vector regression". In: *Statistics and computing* 14.3 (2004), pp. 199–222.

[53] Slawek Smyl. "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting". In: *International Journal of Forecasting* 36.1 (2020), pp. 75–85.

[54] César R Souza. "Kernel functions for machine learning applications". In: *Creative Commons Attribution-Noncommercial-Share Alike* 3 (2010), p. 29.

[55] Robert Stackowiak, Joseph Rayman, and Rick Greenwald. *Oracle data warehousing & business intelligence SOqlos*. John Wiley & Sons, 2007.

[56] Jimin Tan et al. "A critical look at the current train/test split in machine learning". In: *arXiv preprint arXiv:2106.04525* (2021).

[57] Sean J Taylor and Benjamin Letham. "Forecasting at scale". In: *The American Statistician* 72.1 (2018), pp. 37–45.

[58] Guido Van Rossum. *The Python Library Reference, release 3.8.2*. Python Software Foundation, 2020.

[59] Mahendran Venkatachalam. *Recurrent Neural Networks*. https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce. [Online; accessed 25-June-2021]. 2019.

[60] Peter R Winters. "Forecasting sales by exponentially weighted moving averages". In: *Management science* 6.3 (1960), pp. 324–342.