NATIONAL ENGINEERING SCHOOL OF TUNIS

University of Tunis El Manar, Tunisia

Industrial Engineering Department

Year-end project II

Demand Forecasting

Developed by :

MASMOUDI Imen, GHAZOUANI Imen

2^{nd} year Industrial Engineering

Supervised by : Mrs. MEJRI Imen , Mrs. BHAR LAYEB Safa

Members of the Jury:

President: Mrs. CHAABOUNI Fatma

Examiner: Mrs. SOLTAN ALLANI Nada

Academic year: 2021/2022

Acknowledgements

With these few words, we would like to pay tribute to all the people who have contributed to the good progress of this project and its accomplishment in the best conditions.

This work would not have been possible without the presence of our supervisor Mrs. Safa Layeb and Mrs. Imen Mejri, our professors at the ENIT who were always motivating, helpful and responsive. Their valuable advice had spared no effort to transmit to us information and put us on the right track.

We would also like to thank the members of the jury Mrs Chaabouni Fatma and Mrs Soltan Allani Nada for having devoted their precious time to judge this work.

Abstract

In our work we started with a research about the classical statistical methods used for forecasting, and then we started exploring the sales data provided by the company. Afterwards, we went to the modelling part using a Deep Learning approach, where we prepared the data and the model and tuned the hyper parameters and then started the training process.

After that we made some predictions for the year that followed the years given and we compared the predictions with the models that were already made last year by our colleagues.

Key words: Demand forecasting, Machine Learning: ML, Deep Learning: DL, Neural Network Method

Résumé

Dans notre travail nous avons commencé par une recherche sur les méthodes statistiques classiques utilisées pour la prévision. Après nos recherches, nous avons opté pour une nouvelle approche de prédiction basée sur le Deep Learning. Pour ce faire, nous avons commencé à explorer les données de vente fournies par la société : Henkel. Et puis nous sommes passés à la partie modélisation où nous avons préparé les données ainsi que le modèle puis nous avons défini les hyper paramètres, pour enfin démarrer le processus d'apprentissage.

Après cela, nous avons fait des prédictions pour l'année suivante des années déjà présentes dans la data et nous avons comparé les prédictions avec les modèles qui avaient déjà été faits l'année dernière par nos collègues.

Mots clés : Prévision de la demande, L'apprantissage de la Machine : ML, L'appretissage Profond de la Machine : DL, Méthode de réseau des neurons

Contents

Acknowledgements I

Abstract I

Résumé I

List of Figures III

List of Tables V

List of Abbreviations VI

General Introduction VII

1 Literature overview 1

1.1 Introduction . . . 1
1.2 Overview of Demand Forecasting . . . 1
1.2.1 Demand forecasting definition . . . 1
1.2.2 Demand forecasting is the key to an organization . . . 1
1.2.3 Different Types of Demand Forecasting . . . 2
1.2.4 The stages of forecasting . . . 3
1.3 Methodologies . . . 3
1.3.1 Qualitative Techniques . . . 5
1.3.2 Quantitative Techniques . . . 8
1.3.3 Forecast errors . . . 15
1.4 Conclusion . . . 16

2 Deep Learning-Based Demand Forecasting Models 17

2.1 Introduction . . . 17
2.2 Case of study . . . 17
2.2.1 Our mission . . . 17
2.2.2 Used Tools . . . 18

2.2.3 Work Overview 20
2.3 Data Pre-processing 20
2.4 Visualisations 23
2.4.1 Sales' visualisations 23
2.4.2 Normalized data visualisations 25
2.4.3 Correlations' matrix 27
2.4.4 Interpretation 29
2.5 Deep Learning Proposed Models 29
2.5.1 CNN for Time Series Forecasting 29
2.5.2 LSTM for Time Series Forecasting 31
2.6 Conclusion 33

3 The Proposed Hybrid Deep Learning Model 34
3.1 Introduction 34
3.2 The CNN-LSTM Architecture 34
3.3 CNN-LSTM Training and Prediction Process 37
3.4 Modelling 40
3.4.1 Model's Architecture 41
3.4.2 Tuning the Loss Function 42
3.4.3 Tuning the number of epochs 42
3.4.4 Setting the learning rate 44
3.5 Fitting the Model 45
3.5.1 Training the model 45
3.6 Making Predictions 46
3.7 Comparing the loss with the ARIMA and LSTM models 47
3.7.1 The results of all of the models 47
3.7.2 Discussion 50
3.8 Conclusion 50

General Conclusion 51

Bibliography 53

List of Figures

1.1 Forecasting Roadmap 3
1.2 Methodology Tree for Forecasting 4
1.3 Classification of traditional methods 10
1.4 Random Forest Structure 14
1.5 Single hidden layer ANN 15
1.6 Forecast Errors 16

2.1 Connecting to the drive and reading the csv 21
2.2 Hands on the data 21
2.3 Data Pre-processing 22
2.4 Numerizing the Data 22
2.5 Code for Data Visualization 23
2.6 Products' names 24
2.7 Sales Data 24
2.8 Sales Visualization 25
2.9 Normalized Data 26
2.10 All the products sales' normalization 27
2.11 Correlation Matrix 28
2.12 Correlation Matrix Heatmap 28
2.13 The CNN's Structure 30
2.14 The LSTM's Structure 31

3.1 Hybrid Method structure 35
3.2 CNN-LSTM architecture with two convolutional layers, a pooling layer, an LSTM layer, and an output layer 36
3.3 Hybrid Method structure 37
3.4 CNN-LSTM Training and Prediction Process 39
3.5 Windowing the dataset 40
3.6 Model's Architecture 41
3.7 Epochs number for the mse 43
3.8 Epochs number for the Huber 43

3.9 Accurate Learning Rate 44
3.10 Code for setting up the Learning Rate 44
3.11 The Learning Rate Evolution 45
3.12 Making Predictions 47
3.13 Comparison for February, March and April 48
3.14 Comparison for May 49
3.15 Comparison for June and July 49

List of Tables

1.1 Forecast Errors. 16
3.1 Comparison For February, March and April 48
3.2 Comparison for May 49
3.3 Comparison for June and July 49

List of Abbreviations

ML : Machine Learning

DL : Deep Learning

ANN: Artificial Neural Network

LSTM: Long Short Term Memory

CNN : Convolutional Neural Network

MA : Moving Average

ES : Exponential Smoothing

ARIMA : Auto-Regressive Integrated Moving Average

SARIMA : Seasonal Autoregressive Integrated Moving Average

K-NN : K-Nearest Neighbor

MdAPE : Median Absolute Percentage Error

RMSE : Root Mean Square Error

MAE : Mean Absolute Error

Colab : Google Colaboratory

GPU : Graphics Processing Unit

EDA : Exploratory Data Analysis

General Introduction

Sales without demand forecasting are like a fish without water!
One of the most important questions that producers asks themselves when producing is: How much should we produce next? That is when we see and feel the importance of forecasting your demand.

Among the primary goals of demand forecasting are inventory management, as well as the fact that demand forecasting provides very helpful information for all supply chain activities.

Forecasting is used to make decisions about purchasing, marketing, manufacturing, management, budget, and logistics. In the short, medium, and long term, demand estimation is essential. Sales analysis is utilized at the operational level to make purchasing, production, inventory management, and truck routing choices. Demand forecasting is used in the medium term to plan logistical operations and identify the resources required for manufacturing, distribution, and transportation. Demand forecasting is used in the long term to make strategic decisions about new product development, outsourcing, and so on.

In this context, the objective of this end-of-year project is is to forecast the demand for Henkel, a company specialized in detergents and cleaning products, by using the tools of Machine Learning and Deep Learning, Artificial Neural Networks.

In our work, we will be doing demand forecasting with a model that contains neural networks and recurrent neural networks. We want to create a model that enables the enterprise Henkel to get accurate predictions of the upcoming sales demand.

The work accomplished is presented in this report in three chapters. In the first chapter we did some research about demand forecasting and the classical methods that can be used. Then in the second chapter, we did the data pre-processing part where we explored the information we have. And then in the final chapter we went to the last part which was modelling and getting predictions, and comparing them to the real values and comparing them to the predictions released by the other models created by our colleagues last year using the ARIMA and LSTM.

Chapter 1

Literature overview

1.1 Introduction

This current chapter is dedicated to giving a global overview of the demand forecasting concept.

The first part presents the concept of demand prediction, including its definition, its historical evolution, the advantages and limitations behind it, while the second part delves into the many ways used to implement it.

1.2 Overview of Demand Forecasting

1.2.1 Demand forecasting definition

Generally, demand forecasting is the process of predicting and estimating future demand for certain products or services based on historical data, already observed past events as well as future events when known in advance.
It is an approach widely studied and used in many sectors: medical, industrial and commercial.

1.2.2 Demand forecasting is the key to an organization

Forecasting demand for a product or service has proven to be a critical component of any small or large company. The planning of their exercises improves significantly when using this technique.

Indeed, in a finance unit, for instance, the use of demand forecasting provides visibility into future sales and operational expenses. Thus, will have good budgeting that will be as faithful and reliable as possible to the reality of the expenses.

On another level, the use of demand forecasting in logistics makes it possible to manage stock, thus avoiding overstocking or shortages, major challenges for suppliers and any industrial company. A fairly significant power gained is that of ensuring a satisfactory customer delivery time.

As for the use of this approach in marketing, it is mainly aimed at developing a good marketing strategy for management. With an understanding of the demand, marketing campaigns will be targeted at specific times of the year and for a specific product/service. It also allows you to plan the use of promotional coupons.

1.2.3 Different Types of Demand Forecasting

Forecasting plays a crucial role in decision-making for companies and helps them see significant improvements in both productivity and profitability. Some firms need short-term forecasts, others medium-term and others long-term forecasts. It all depends on their sectors of activity and the strategy of their business.

1. **Short-term forecasts:** The adequate period for such forecasts is generally less than 12 months (We are talking about a few weeks, months, the most common is 6 months). Companies using this type of forecast seek to:

   (a) Optimize staff planning (Manage the need to hire new experts / Dismissing others / set well-defined working times for the entire period to come) Control of maintenance expenses

   (b) Ensure that the production line runs continuously, thus ensuring an uninterrupted supply of products

   (c) Development of sales strategies

2. **Medium-term forecasts:** Commonly, required to determine future resource needs that will allow the company to size its stock as well, optimize its purchases of raw materials, and identify its supplies in terms of maintenance means, production machinery, and other equipment.

3. **Long-term forecasts:** This type of forecast is made for periods that exceed 12 months. The major interests behind the use of this type are:

   (a) Annual strategic planning: This case of decisions also requires heeding market opportunities, environmental factors, and the internal environment of the company.

   (b) Long-term financial planning, indebtedness, and acquisition of funds

(c) Decision-making regarding business expansion

1.2.4 The stages of forecasting

The figure below shows the forecasting process, from data gathering to deployment of the forecasting model. The first step, as has been said, relies on the collection of adequate data. Today, this milestone is enhanced by new techniques (Data mining, Big data...) to ensure that the data will deliver consistent and accurate results. Subsequently, different methods can be applied to the forecast. Indeed, one quantitative or qualitative forecasting technique will be used to calculate the estimates.

Once these values have been reviewed to ensure that they make sense to the company, either an action is taken or the method is reviewed if the results are not conclusive.

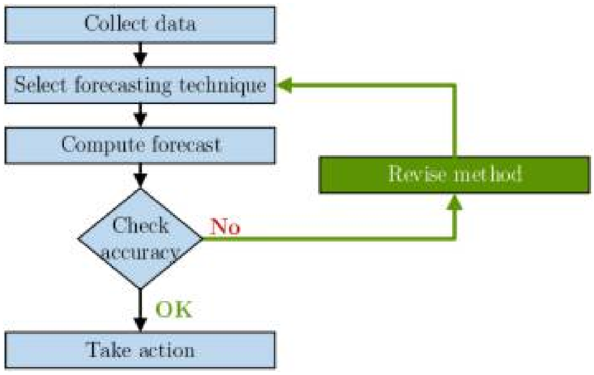

Figure 1.1: Forecasting Roadmap
[4]

1.3 Methodologies

Over the last seven decades, demand forecasting has been the most sought-after approach to optimization in all business lines. Several studies have been conducted on this subject investigating the different possible techniques to estimate the demand in the future.

The choice of the appropriate technique depends mainly on the available data (in terms of quality, which is an essential element of an accurate demand forecast, as well as in terms of the range of periods over which the data have been collected).

Indeed, for example, forecasting the demand for a new product in a company that has no data on it, or forecasting the demand for a product that has just been launched, so we only have data for a few months or even weeks, is done using techniques that differ from forecasting the demand for an old product for which we have much more information over a longer period (years).

In this section, we provide an overview of possible methods that could be used to forecast demand and the available evidence on their use. They are mainly divided into two main families: quantitative methods and qualitative ones.

The figure below sums it all up and the following parts will explain some of the techniques in more detail.

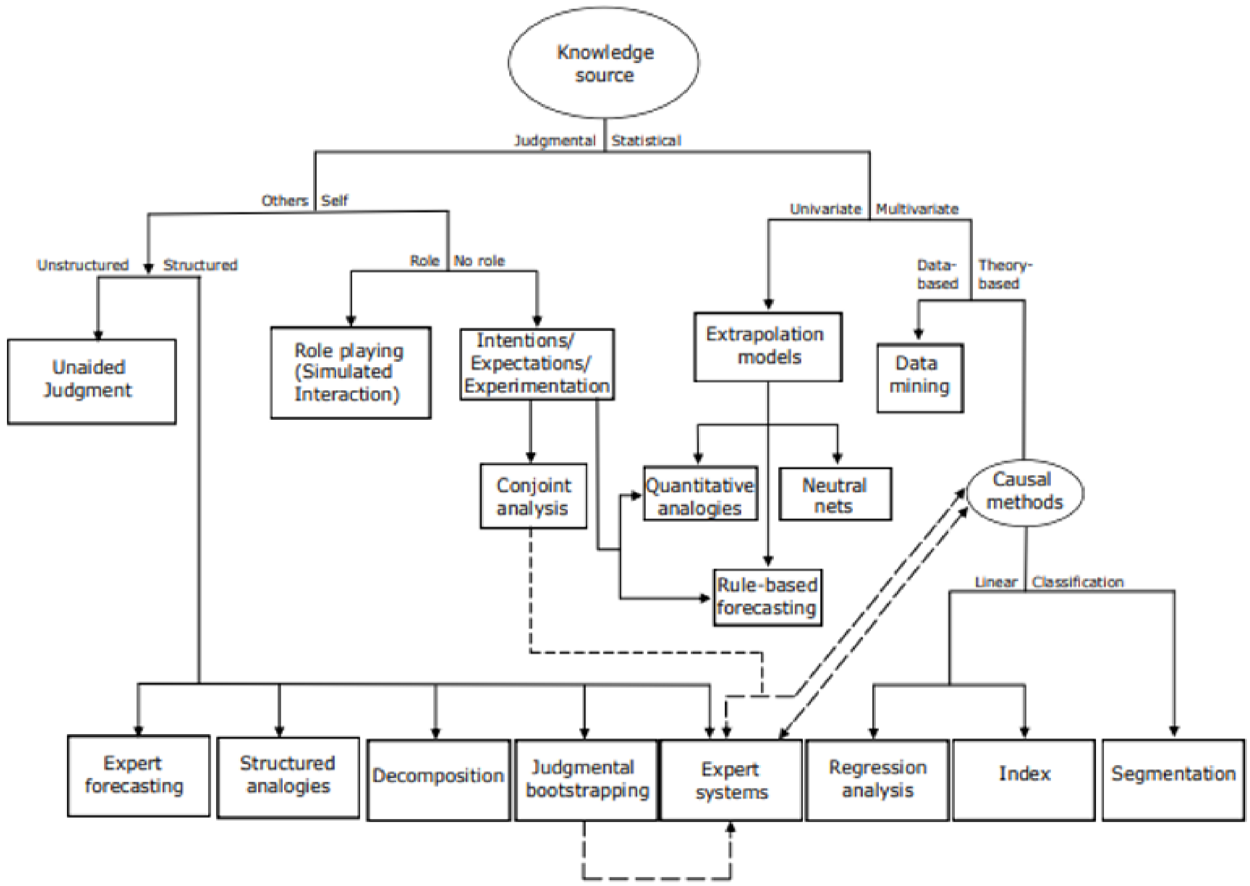

Figure 1.2: Methodology Tree for Forecasting
[4]

1.3.1 Qualitative Techniques

These techniques are most often used in cases where there is no data available or if the quality of the available data is so poor that it is not relevant for prediction. We may also use this technique when a new competitor enters the market and we have no historical precedents as to its effect on the business of the company.

Unlike other methods, these techniques do not always take historical data into consideration when generating forecasts, they rely basically on experts' judgment.

In addition, experts are often subject to bias when making forecasts. Therefore, the strategy of the following qualitative techniques is to use a heterogeneous group of experts in the hope that their differing biases will tend to cancel one another.

1.3.1.1 Judgment forecasts Methods

- **The Delphi method**
  This technique was invented in 1950 by Olaf Helmer, Norman Dalkey, and Rand Corporation as a remedy to certain military problems [12].
  This approach is based on a key idea that is: The opinion of a group in forecasting is more consistent and closer to reality than the opinion of a single individual. Therefore forecasting with Delphi necessarily requires a group of experts, with a number that ranges from five to twelve persons, with diverse backgrounds to make consensus forecasts in a structured and iterative way. In a sense, the Delphi method is a controlled debate.The following steps are the stages for performing a Delphi method:

  1. Gather experts.
  2. Clearly and concisely define the forecasting task..
  3. Preliminary judgment of the experts and justification of the information gathered within the group. Afterward, feedback to the experts will be made including numerical data summaries and a graphical representation derived by a facilitator.
  4. Launch a second round, much better than the first, because experts have altered their estimates in response to criticism. And so on, until a satisfactory final decision is achieved, this phase will be performed multiple times.
  5. The final forecasts are created by combining the forecasts of the experts.

  Although this method is relatively inexpensive since the panelists do not meet in the same place which avoids the costs of assembling a group of highly-paid individuals, the limitation of such a method remains in the consumption of time [6].

- **Forecasting by analogy**
  It is similar to the Delphi method; the difference is in some additional steps that follow the step of getting the first feedback. Instead of reviewing it and launching a second conference, experts will identify and describe numerous analogies and generate predictions based on those analogies. Right after, like Delphi, the forecast passes on to the decision-makers of the organization.

  Let's take the example of the prediction of the demand for a new product, the experts look for a similar product in the market for which they have the prediction and make assimilation (**analogous products**). The collation for similarities is done mainly based on attributes (features) that experts consider important.

  Overall, the analogous products method may be a tool that permits forecasting demand for new products. As a first step, the forecaster starts with collecting analogous products, then proceeds to their examination [8].

  Once the examination is done, forecasting the demand for new products can be performed based on the patterns identified in the first step.

  Even though the accuracy of this approach's forecasts is still questionable, the evidence on structured analogies for forecasting decisions in conflicts [13], suggests that the forecasts would be useful.

  In fact, large errors are common in new product forecasts. Tull (1976) estimated the mean absolute percentage error for new product sales to be around 65 percent [31].

  Extrapolation techniques are effective to utilize once a new product is on the market. The selection of the proper functional form has been highly considered (with attention) for early sales. Two different types of literature exist:

  - The diffusion literature uses an S-shaped curve in order to predict new product sales. The curve builds up slowly in time and increases rapidly when buyers become more and more interested in the product. In other words, when a good word of mouth is circulating. The sales growth keeps on increasing until reaching a saturation level.

- The substantial literature determines which model fits best the sales trend. However, only a few studies revolve around comparative validation [24].

- **Scenario forecasting (Simulation)**
  Simulation techniques attempt to mimic the demand by combining causal and time series methods which will be explained later on in this chapter [7]. The scenarios are generated by considering all possible factors or drivers, their relative impacts, the interactions between them, and the targets to be forecast.

- **Judgmental bootstrapping**
  It is a method, generally used in the case where the data available is poor in terms of quality, it makes it possible to develop a clear, simple, and complete expert system which translates the rules imposed by the predictors into a quantitative model [3].

1.3.1.2 Limitations of judgment forecasts

- Implementation time [28]:
  This technique takes longer than other math-based ones to be implemented correctly. It takes a few months while others last one to a few days.

- The cost of deploying these techniques is quite high

- Accuracy [20]:
  Judgment-based predictions are certainly subjective, therefore they cannot be free from bias or limitations.
  The inconsistency of this approach seems to be questionable / not truly accurate since it relies on human knowledge, senses, and experience, which leads us to the fact that this method is limited by human error. For instance, the human mind tends to give more importance to recent events than past events. Setting the last observed value as a reference point can drive to a loss of crucial information and thereby bias the results. This phenomenon is called anchoring. Moreover, human decisions are largely influenced by psychological factors. A positive or negative state of mind generates optimistic/pessimistic forecasts.

1.3.1.3 Improving judgment forecasts

- **Set rules**
  In order to improve the consistency of judgmental forecasts, documenting the decision rules may be an appropriate solution. It will help predict the demand while respecting certain norms. Furthermore, asking a forecaster to justify their forecasts enhances accountability, so formal documentation reduces bias.

- **Implement a systematic approach**
  Well-structured and systematic approaches (statistical forecasts: these methods are discussed in part 3 of this chapter) can be combined with a judgmental forecast in order to boost its reliability, but this is in particular cases. There are several strategies to do so. Selecting the proper one depends on the forecast conditions and mostly works in the environment where the forecasters have high experience with the products and markets. The fact of combining improves accuracy was highly recommended by Armstrong and Collopy (1998) that summarized research in this area [2].

1.3.2 Quantitative Techniques

Turning now to situations where there is ample data (numerical information about the past) as well as it is obvious that some forecasts follow the same past patterns, quantitative methods are the appropriate choice for prediction.[30]

Depending on the discipline and the goals behind the demand forecasting tasks, an important number of quantitative forecasting methods exist. In fact, each one of them is characterized by its own features, accuracies, and implementation costs. Choosing the proper method implies also taking into consideration all these factors/elements in addition to the relationships between causes and effects. It is, actually, highly important to keep an eye on these relationships. In other words, if the knowledge of them is poor or insufficient, considering the kind of available data becomes a must.

Most quantitative prediction tools rely on either time-series data (collected at regular intervals over time) or cross-sectional data (collected at a single point in time).

In this current report, we are selecting and focusing on forecasting future data. More specifically, our main focus is on the time-series domain.

1.3.2.1 Time Series Approach explanation

When data is observed sequentially over time, we consider what we call the time series.

In this current report, only time series that are observed at regular intervals of time are considered (hourly, daily, weekly, monthly, annually). More specifically, in our case, the monthly interval. Irregularly spaced time series can also happen but they are beyond the focus of this coding and research project.
Forecasting time series data aims at estimating the continuity of the sequence of observations in the future.

The historical data includes trends, cyclical fluctuations, seasonality, and behavior patterns.

The variable to be forecasted is the main information used when it comes to the simplest time series forecasting methods. In other words, there is no taking into consideration of research and/or exploration of the factors behind its trends and behavior.
Consequently, these trends and seasonal patterns will be extrapolated over other types of information (such as marketing, market competition, and so on).

As previously mentioned, words/terms such as "trends" and "seasonal" need to be detailed and expressed differently:

- **Trend:**
  When there is a long-term increase or decrease in the data, there is what is called "a trend". In fact, it does not have to be necessarily linear but, sometimes, it might change direction from an increase to a decrease.

- **Seasonal:**
  A seasonal pattern happens when a time series is affected by seasonal factors. For instance, when it is influenced by the time of the year or the day of the week. In fact, seasonality is always of a fixed and known frequency.

- **Cyclic:**
  A cycle occurs when the data shows increases and decreases that are not of a fixed frequency. These fluctuations are usually a result of economic situations, and are usually in relation to the "business cycle". Most commonly, the fluctuations' duration is a minimum of two years.

When picking/choosing a forecasting method, first, identifying the time series patterns in the data is needed. That is followed-up with choosing the right/proper method which is able to spot the patterns properly.

1.3.2.2 Traditional Methods

The traditional methods are mainly divided into two families as shown in the figure below:

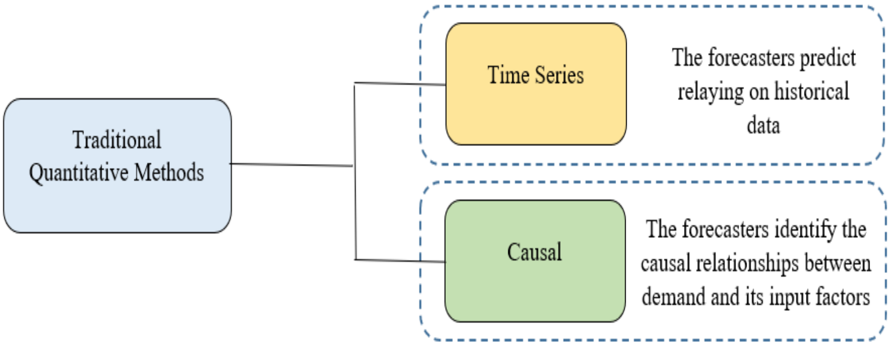

Figure 1.3: Classification of traditional methods
[10]

1.3.2.2.1 Statistical Time Series Approaches

The generation of a ‘rolling average’ for future demand is, often, created by statistical time series methods using historical data.

More sophisticated methods use weighted data so that more recent data has more of an effect than older data. These tools are usually less expensive and faster to put in place than other forecasting methods. But, that does come with lower accuracy than more costly and time-consuming alternatives

As a consequence, statistical time series methods are best suited for:

- **Mid to long-term forecasting**
- **Products that are well-established and have stable demand**
- **Predicting total demand rather than for individual products** [32].

A) **Moving Average (MA):**
The author Nguyen and the author Hansun have done in 2010 extensive research on the quality of MA outcomes [14].

This method entails calculating a tribe/set of averages, each of which corresponds to a well-defined value of the trend for a certain time period or interval [5]. To be more precise, the averages generated each time serve as the foundation for generating all-new data for the same period.

The term "moving average" refers to calculating the average while replacing the previous value and showing the data from the next point. This approach is completely self-contained.

This method is entirely dependent on mathematics (mathematical convolution [17]).

This is also how the moving average of order m, where m is odd, is written:

$$\hat{Z}_t = \frac{1}{m}\sum_{j=-k}^{k} y_{t+j} \;;\quad \forall\ t = k+1, \dots, n-k \qquad (1)$$

- m=2k+1: moving average of order m. Also, known as 'm(odd) - MA'
- n : total number of data points used.

On the other hand, for the case where m is even, the calculation is as follows:

$$\hat{Z}_t = \frac{1}{m}\sum_{j=0}^{k} y_{t-j} \;;\quad \forall\ t = k+1, \dots, n \qquad (2)$$

- m=k+1 moving average of order m. Also, known as 'm(even) - MA'
- n total number of data points used

In the second instance, that of (m-even), it is critical to align the current averages in the middle of the relevant data that is being averaged. Otherwise, analyzing trend lines will become quite tough. The 'centered moving average,' or 2 × m (even) MA, is the name for this method. You should also be aware that smoothing a moving average with another moving average is an option. A twofold/ double moving average is the name for this method [18].

The moving average can consider each previous period equally or unequally, and this is known as the weighted moving average.

B) **Exponential Smoothing (ES)**:
Exponential smoothing computes weighted averages of historical observations,

where the weights decline exponentially over time. This method prioritizes or emphasizes the most recent observations [19]. This framework generates accurate predictions in a variety of time series in a short amount of time. This is a significant benefit that is critical for industrial applications.
For this type of method we mainly distinguish:

- Simple exponential smoothing
- Double exponential smoothing
- Triple exponential smoothing

Recognizing the key components of the time series (whether it's a trend or seasonal) as well as the way they enter the smoothing method is necessary in order to choose the best approach among them (e.g., in an additive, damped, or multiplicative manner).

C) **Auto-Regressive Integrated Moving Average (ARIMA)**:
Unlike exponential smoothing models which describe the trend and seasonality detected in the data, the ARIMA model treats the autocorrelations in the data, i.e. the linear relationships between the lagged input variables, and describes them very judiciously.

D) **Seasonal Autoregressive Integrated Moving Average (SARIMA):**
An ARIMA model's extension that supports uni-variate time series data addressing backshifts of the seasonal period.

Limitations:
Because time series approaches require data that is stable - meaning, data that follows predictable trends - they can fail to foresee market saturation of a product, as well as sudden spikes in demand and seemingly inexplicable shifts in client preferences [35].

1.3.2.2.2 Causal Models (Multivariate)

The causal techniques create a forecast by establishing a link between demand and any other external or internal factors that may have influenced it. Given a sound theory and a few critical variables, econometric approaches are useful among causal methods. When there are a lot of variables and a lot of information about the scenario, index models come in handy.

1.3.2.2.3 Data Mining

Data mining, is an analytical procedure for extracting useable data from big datasets or unprocessed data in order to detect possible patterns and trends [33].

(1) Selection, (2) pre-processing, (3) transformation, (4) modeling, and (5) interpretation are the five sections of the process. Items (1), (2), and (3) deal with determining the target values required for the problem's scope and converting them into a standard format.
The fourth step is to utilize algorithmic approaches to fit and uncover hidden patterns in the processed data. Finally, Step (5) focuses on extracting model outputs in order to provide more feedback on problem solutions and knowledge-driven decisions. Organizations are prepared to invest money in data mining since it is an effective tool.

1.3.2.3 Modernizing Demand Forecasting with Machine Learning

1.3.2.3.1 K-Nearest Neighbor (K-NN) Regression

The K-NN technique compares the training set to new, unseen data and generates a prediction based on determining the shortest distance between training data points, often known as their "nearest neighbor," and the point where the forecast is to be made.

1.3.2.3.2 Ensembles of Decision Trees (Random Forests)

Models for classification and regression procedures such as decision trees are well-known. In both regression and classification, the decision-making process involves a series of if/else questions. The fundamental issue with decision trees, however, is that they have a tendency to overfit the data [25].

A back-propagation network is used to recognize handwritten digits.
To try to tackle this problem,ensembles of decision trees were built. Random forests and gradient-boosted decision trees are the two most used techniques.

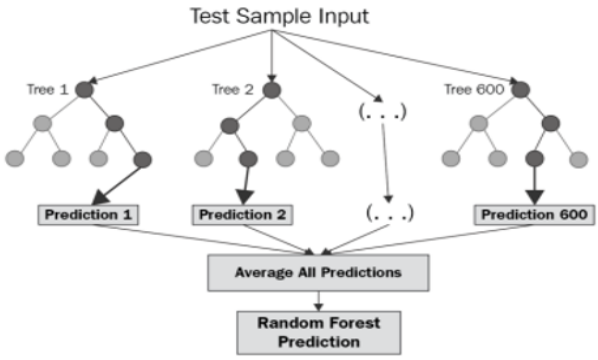

Figure 1.4: Random Forest Structure
[25]

1.3.2.3.3 Artificial Neural Networks models (ANN) Models (Deep Learning)

Artificial neural networks (ANNs) are designed to emulate the human brain's data processing capability. The human brain is composed of billions of linked neurons, each of which stores little quantities of data. However, by connecting them, vast and complicated amounts of data may be stored and handled. An artificial neural network is developed to predict time series using a number of historical observations as input. To minimize the overall prediction error, several runs are undertaken to update the weights that decide the output result. In ANNs, a complicated nonlinear relationship between input and output can be created [34].

ANNs, unlike exponential smoothing models, are not constrained by data patterns such as trends or seasonality. It can even represent nonlinear data patterns and learns these patterns straight from the data. In our complex world, interdependence is rarely linear; in fact, nonlinearity may be found in many real-world projections. Any continuous linear and nonlinear function can be approximated with any level of precision ([39],[16]). Adaptive linear neurons, multi-layer perceptrons, and radial basis functions are examples of neural network structures that have been given in the literature. The majority of research, however, employs a multilayer perceptron structure to forecast demand [11].

A three-layer neural network was proposed to predict. The back-propagation algorithm teaches it an input layer structure, hidden layers, and an output layer structure.

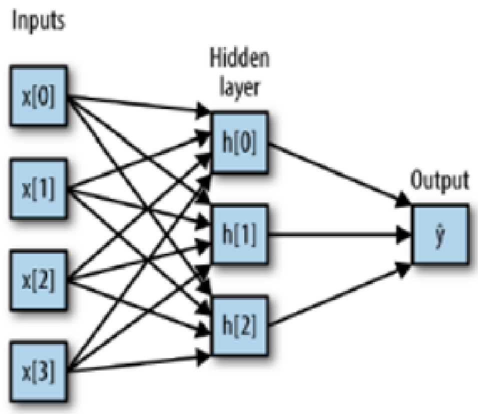

Figure 1.5: Single hidden layer ANN
[11]

The information collected from the input neurons is multiplied by its allotted weight during training. When the result passes a particular threshold, the activation function is triggered, and the signal is delivered to the output neurons. The prediction error is calculated and returned to the neural network after each round.

When utilizing artificial ANNs to predict, the structure must first be defined, followed by the training steps and the determination of the appropriate weights. The number of input, output, hidden layers, and the relationship between nodes transfer learning Schemes function, training, and testing samples, training method, and performance metrics must all be determined before the structure can be determined [11].

Backpropagation training performed far better than other training methods, according to Kumar and Herbert [9].

The amount of hidden variables can affect the performance of ANNs. To deal with nonlinearity, Zang found a sufficient number of hidden layers [36].

1.3.3 Forecast errors

Forecast error assessment is useful for selecting the proper forecasting model and assessing the required precautionary reserves, in addition to providing significant information about the amount and unpredictability of random components. Table 1 shows the most common forecasting errors, how to compute them, and the allowed intervals.

Error Name	Formula	Description
The Root Mean Squared Error (RMSE)	$RMSE_t = \sqrt{\sum_{i=1}^{t} (E_i - D_i)^2 / t}$	The smaller the better
The Mean Absolute Deviation (MAD)	$MAD_t = (\sum_{i=1}^{t} \lvert E_i \rvert) / t$	The smaller the better
The Mean Absolute Percentage Error (MAPE)	$MAPE_t = (\sum_{i=1}^{t} [\lvert E_i / D_i \rvert \times 100]) / t$	≤ 50
The Mean Percentage Error (MPE)	$MPE_t = \sum_{i=1}^{t} [E_i / D_i] / t$	The smaller the better
Bias	$bias_t = \sum_{i=1}^{t} [E_i]$	Smaller numbers are better
Theil's U	Statistics	≤ 1
Ljung-Box Q	Statistics	Applicable in ARIMA model
Tracking Signal	$TS_t = bias / MAD_t$	Between -6 to +6

Table 1.1 : Forecast Errors
[36]

When few series are available, the Median RAE (MdRAE) is advised for selecting the most accurate methods, while the Median Absolute Percentage Error (MdAPE) is preferred otherwise. Because the Root Mean Square Error (RMSE) is unreliable, it should not be used to compare accuracy across series.

1.4 Conclusion

In this study, a combination of artificial ANNs and time series was used to predict the demand for products and product groups. The goal is to find the best forecast model for each product and product family based on its accuracy.

Chapter 2

Deep Learning-Based Demand Forecasting Models

2.1 Introduction

This chapter focuses more on the practical level. It alternates between literature and experimentation. First, it contains a general overview of our working environment for implementing our prognostications. Next, it contains a brief description of the convolutional, pooling, and LSTM layers that form the core of the proposed model.

2.2 Case of study

Over the past half-century, forecasting has been witnessing paramount advances. These advances can be exploited to enhance many aspects of forecasting demand. Most recently, gains have emanated from the integration of advanced data science techniques to forecast customer demand. Thereby, much about how to gain acceptance of forecasts was learned.

2.2.1 Our mission

In line with previous studies on demand forecasting for a product or a group of products, Henkel, a global brand that works on a different axis among them is the production of detergent products, has been chosen as the data source for this study.
Henkel classifies its detergent products into four categories:

- Nadhif HS REG (They provide packages of 320g)
- Nadhif HS MS (They provide packages of 320g)
- Dixan LS Reg (They produce two types of packages: a 4Kg one and a 5Kg one)
- Dixan LS SDM (They produce two types of packages: a 4Kg one and a 5Kg one)

Results based on their real sales data sets show that demand forecasting for their detergent products is a complex and non-linear problem and that our mission is to propose an approach using advanced machine learning techniques that can predict their product demand with high accuracy.

2.2.2 Used Tools

2.2.2.1 The use on Machine Learning

Machine Learning is a sub-set of artificial intelligence where computer algorithms are used to autonomously learn from data and information. In machine learning computers don't have to be explicitly programmed but can change and improve their algorithms by themselves.
Today, machine learning algorithms enable computers to communicate with humans, autonomously drive cars, write and publish sport match reports, and find terrorist suspects. I firmly believe machine learning will severely impact most industries and the jobs within them, which is why every manager should have at least some grasp of what machine learning is and how it is evolving.
The reason why artificial intelligence took so much time to appear in the practical world is because: the amount of calculus acquired very strong compute power, and this latter took some time to appear.
Yet now, when we talk about the complexity of the calculus behind The models created in Machine Learning and in Deep Learning, we feel like we have a long way to go in improving our compute power! ML and DL demand enormous computational power because of the enormous amount of mathematical operations so that it can optimise the model's weights as it learns or let's say gets closer to reality and gives better predictions and results.
That is why we are using Artificial Neural Networks for our work, because it has been proven that it is evolutionary when it comes to looking for pattern and predicting the future events.

2.2.2.2 The utilization of Python and Google Colaboratory (Colab)

A) **Why choose Python for such a forecasting task?**

First, of all, Python is a popular general-purpose programming language that can be used for a wide variety of applications. It was created by Guido van Rossum, and released in 1991.
It's one of the most popular and useful programming languages since it's very attractive for Rapid Application Development due to:

- It's simple and easy syntax.
- Its support modules and packages.

In this and the following chapter, we will perform forecasting with deep learning algorithms by implementing them with python.

B) **Why coding on Google Colab?**

Google Colaboratory is the impeccable Chromebook coding experience to utilize Python (especially for machine learning). It is an executable document that lets you indite, run, and share code within Google Drive.
A notebook of Colab is composed of cells where each one can contain code, text, images, and more. Colab connects the notebook to a cloud-based runtime, meaning the user can execute Python code without any required set up on his own machines.
Supplemental code cells are executed using that same runtime, resulting in a rich, interactive coding experience in which the coder can use any of the functionality that Python offers.
In the following experimental parts, Colab is the workspace that was utilized!
In conclusion we chose google colab as our working environment for few of the following reasons:

- We can collaborate on the code together on this online environment.
- Here on colab we have amazing features like GPU accelerators, the access we have is still limited but it gives us enough resources to do the work!
- We can link the colab environment to Drive, where we stored our data and where we are intending to store the model h5 file.
- Here we can have code cells and text cells: that enables us to write commants on the code created, which is very helpful when it comes to collaborating and to having reminders to yourself on what you were thinking when writing the code.

2.2.2.3 Essential Libraries

In order to execute the deep learning algorithms, essential libraries have to be used:

- **Tensorflow:** It's an open-source framework. Pristinely developed by the Google Encephalon Team within Google in 2015. It's a bunch of APIs developed to build ANN models.

- **Keras:** It's a high-level neural network library and it's tyro-convivial since it avails building training models in a facile way.

2.2.3 Work Overview

In our pipeline we started with the EDA: Exploratory Data Analysis part where we imported our data, pre-processed it and visualised and analysed it. And then went to the modelling part where we prepare the data for the model and prepared the model and then fitted it to the data, and tuned the parameters

2.3 Data Pre-processing

After seeing the first 12 rows of our data, we noticed that there were unwanted commas that separated the thousands and the hundreds. To remove them we executed this three parts separated code cells and now we have the right number forms we need to move forward.

```
[ ] import os
    os.chdir('/content/drive/MyDrive/PFA||')
    os.getcwd()

    '/content/drive/MyDrive/PFA||'
```

```
[ ] import pandas as pd
    data=pd.read_csv('/content/drive/MyDrive/PFA||/time series henkel Mirror.csv')
```

```
data.head()
```

	Mois_année	P1	P2	P3	P4	P5	P6	Time
0	January'18	32,900	29,458	8,553	6,484	6,825	5,961	1
1	February'18	26,429	11,128	8,632	7,894	5,840	8,541	2
2	March'18	23,141	31,258	4,162	7,731	5,467	8,034	3
3	April'18	23,680	25,211	3,348	2,235	6,548	2,583	4
4	May'18	31,372	23,133	4,112	3,069	5,764	1,784	5

Figure 2.1: Connecting to the drive and reading the csv

```
for col in data.columns:
  data[col]=data[col].replace(",", "")
data.head(12)
```

	Mois_année	P1	P2	P3	P4	P5	P6	Time
0	January'18	32,900	29,458	8,553	6,484	6,825	5,961	1
1	February'18	26,429	11,128	8,632	7,894	5,840	8,541	2
2	March'18	23,141	31,258	4,162	7,731	5,467	8,034	3
3	April'18	23,680	25,211	3,348	2,235	6,548	2,583	4
4	May'18	31,372	23,133	4,112	3,069	5,764	1,784	5
5	June'18	25,779	29,350	4,739	5,051	5,877	5,581	6
6	July'18	40,552	29,187	1,838	1,528	1,092	2,649	7
7	August'18	35,702	33,895	3,000	2,711	3,461	7,065	8
8	Sept'18	33,193	35,864	12,659	7,192	8,179	5,037	9
9	Oct'18	31,012	29,751	5,424	2,603	3,528	2,378	10
10	Nov'18	30,560	29,561	3,169	2,224	2,717	3,869	11
11	Dec'18	22,791	15,531	3,079	1,559	3,772	912	12

Figure 2.2: Hands on the data

```
for col in data.columns:
  data[col]=data[col].str.replace(",", "")
data.head(12)
```

```
[ ] for col in data.columns:
      data[col]=data[col].replace(",", "")
    data.head(12)
```

	Mois_année	P1	P2	P3	P4	P5	P6	Time
0	January'18	32900	29458	8553	6484	6825	5961	1
1	February'18	26429	11128	8632	7894	5840	8541	2
2	March'18	23141	31258	4162	7731	5467	8034	3
3	April'18	23680	25211	3348	2235	6548	2583	4
4	May'18	31372	23133	4112	3069	5764	1784	5
5	June'18	25779	29350	4739	5051	5877	5581	6
6	July'18	40552	29187	1838	1528	1092	2649	7
7	August'18	35702	33895	3000	2711	3461	7065	8
8	Sept'18	33193	35864	12659	7192	8179	5037	9
9	Oct'18	31012	29751	5424	2603	3528	2378	10

Figure 2.3: Data Pre-processing

After reading the data and removing the commas, we see that the data's type is string. When plotting the graphs we need numbers that is why we converted the data type to float. Later on we will be doing more complex things with data preparation for the model and that is one other reason why we need them to be numerical.

```
[ ] data["P1"]=data["P1"].astype("float")
    data["P2"]=data["P2"].astype("float")
    data["P3"]=data["P3"].astype("float")
    data["P4"]=data["P4"].astype("float")
    data["P5"]=data["P5"].astype("float")
    data["P6"]=data["P6"].astype("float")
```

Figure 2.4: Numerizing the Data

2.4 Visualisations

2.4.1 Sales' visualisations

The data we have is the monthly sales of six products from January 2018 till January 2021, which makes them 41 values for each product.

```
[ ]  import matplotlib.pyplot as plt
     plt.plot(data['Time'],data['P1'])
     plt.title("Sales Of The First Product")
     plt.show()
     plt.plot(data['Time'],data['P2'])
     plt.title("Sales Of The Second Product")
     plt.show()
     plt.plot(data['Time'],data['P3'])
     plt.title("Sales Of The Third Product")
     plt.show()
     plt.plot(data['Time'],data['P4'])
     plt.title("Sales Of The Fourth Product")
     plt.show()
     plt.plot(data['Time'],data['P5'])
     plt.title("Sales Of The Fifth Product")
     plt.show()
     plt.plot(data['Time'],data['P6'])
     plt.title("Sales Of The Sixth Product")
     plt.show()
```

Figure 2.5: Code for Data Visualization

We renamed the products from P1 to P6 So we can simplify the visuals so here is the correspondence of each product's name in this table:

The first product: P1	Nadhif HS REG 320 g
The second product: P2	Nadhif HS MS 320 g
The third product: P3	Dixan LS Reg 4 Kg
The fourth product: P4	Dixan LS Reg 5 Kg
The fifth product: P5	Dixan LS SDM 4 Kg
The sixth product: P6	Dixan LS SDM 5 Kg

Figure 2.6: Products' names

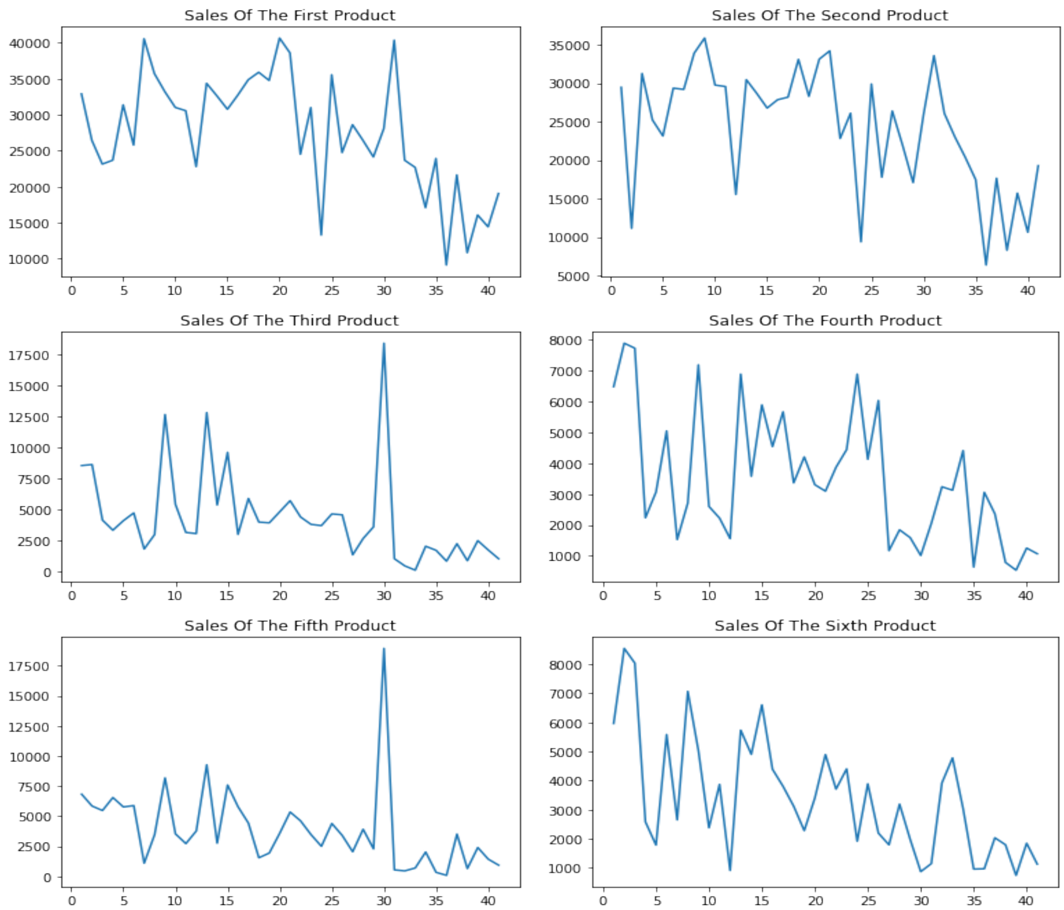

Figure 2.7: Sales Data

Figure 2.8: Sales Visualization

After that we Plotted some figures we can see that the products sales are on different scales, that is why we need to normalize the data so we can make more accurate conclusions.

2.4.2 Normalized data visualisations

After running this code we centralized our products' sales so that they are all at the same scale now.

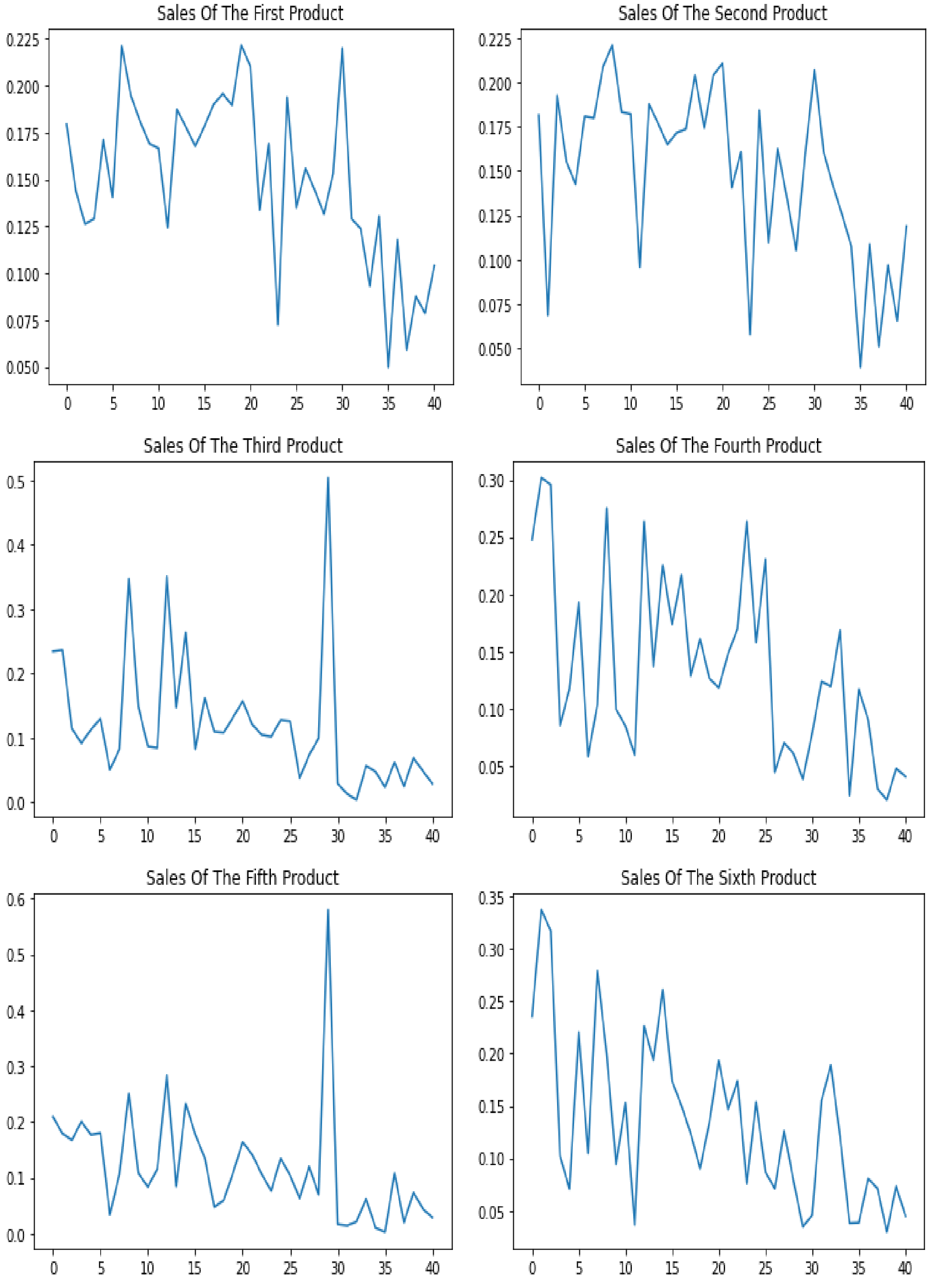

Figure 2.9: Normalized Data

Figure 2.10: All the products sales' normalization

After we normalized the data, we ca see that P3 and P4 seem to have an odd pic around the 28th time step, which happens to be around April 2020, we can explain that by the appearance of the corona virus, so people would buy more detergent to wash their clothes thinking that it will protect them more, or it can be due to a marketing strategy that aims to increase the sales through promotions so people buy more.

Overall, our data seems to be having a downward trend, and that it can get influenced by exterior factors.

2.4.3 Correlations' matrix

Seeing the correlations between features can be one of the best helpers when preparing your model's inputs. If you figured out that for example the day of the week is impacting the sales, you could count it as an input parameter in your model, and then you can let it influence your outputs because in reality, people tend to buy more on weekends. The same goes to pre-events too like in Ramadhane or the Aid, people tend to increase their buying behaviour before these times. After running this code we were able to get the correlation matrix between the six products we have.

```
[ ] df = pd.DataFrame(data,columns=['P1','P2','P3','P4','P5','P6'])
    corrMatrix = df.corr()
    corrMatrix
```

	P1	P2	P3	P4	P5	P6
P1	1.000000	0.859421	0.327932	0.174983	0.247277	0.343429
P2	0.859421	1.000000	0.312991	0.208455	0.292316	0.424289
P3	0.327932	0.312991	1.000000	0.435807	0.903540	0.326260
P4	0.174983	0.208455	0.435807	1.000000	0.296364	0.710488
P5	0.247277	0.292316	0.903540	0.296364	1.000000	0.265497
P6	0.343429	0.424289	0.326260	0.710488	0.265497	1.000000

Figure 2.11: Correlation Matrix

Using Seaborn library helps us visualise the matrix more with the heatmap function.

```
[ ] import pandas as pd
    import seaborn as sn
    import matplotlib.pyplot as plt

    sn.heatmap(corrMatrix, annot=True)
    plt.show()
```

Figure 2.12: Correlation Matrix Heatmap

The correlation matrix shows that there are strong correlations between three couples of products: P1 and P2, P3 and P5 and P4 and P6.

This strong correlation between products can get us deeper, we start getting meaning when we get back to the nature of the products:

- P1 and P2 are both "Nadhif HS Deep Clean 320 Grams" the first is the regular and the second is MS.
- P3 and P5 are both "Dixan LS Powder 4 Kg New" the first is The Regular and the second is MS.
- P4 and P6 are both "Dixan LS Powder 5 Kg New" the first is The Regular and the second is MS.

2.4.4 Interpretation

After seeing the correlations between the products we can conclude that there is a big influence between the couple of products two by two P1 and P2, P3 and P5, P4 and P6. here we can link it to the behaviour of the clients, when the need the product that weights 4 Kg, they will buy of both types for an example because they consider them the same at an extent.
In our model we will include the couples that are correlated because the correlation will help the model to better see the pattern and to give us better predictions.
There are so many factors that can influence your output, that is why we need to take our time when it comes to the first part part of our work in Machine Learning, the data preprocessing, we need to figure out what are the inputs that influence our data the most and then take them into consideration, and that is when the feature engineering comes in handy.
Now that we have analyzed and visualized our data, it's time to get into the deep learning part.

2.5 Deep Learning Proposed Models

In this section, we describe briefly the applied models in this study.

2.5.1 CNN for Time Series Forecasting

2.5.1.1 Learning Schemes: CNN Network Architecture

One of the commonly used deep learning methods, the CNN. A network can contain a range of 1D to 3D CNN (n-D is also used for complex problems).
A 1D CNN is used for sequence data processing. In other words, a 2D CNN is used for image and text recognition, and finally, a 3D CNN is used especially for

medical image and video data recognition.
In the current study, a 1D CNN was adopted. Its process is as described in the figure below:

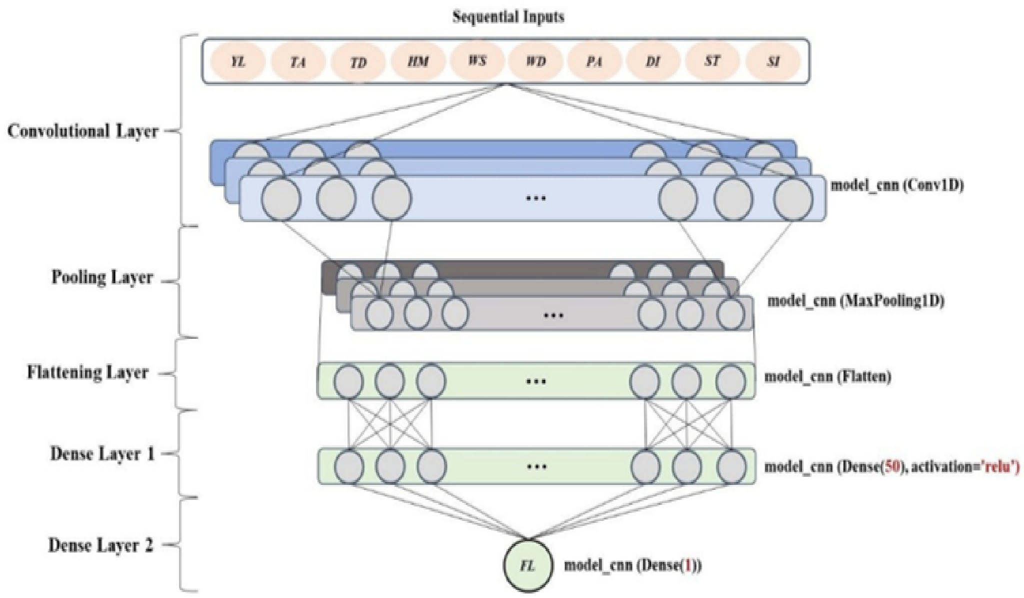

Figure 2.13: The CNN's Structure
[37]

2.5.1.2 Parameters

We mainly have three important parameters in a CNN:

i) **Depth**: It represents the number of filters used for the convolution operation.

ii) **Stride**: It represents the number of pixels by which we slide our filter matrix over the input matrix.

iii) **Zero-padding**: It's often convenient to pad the input matrix with zeros around the border so that we can apply the filter to the bordering elements of our matrix.

2.5.2 LSTM for Time Series Forecasting

2.5.2.1 Learning Schemes: LSTM Network Architecture

LSTM as an extension of RNN has a robust capability in forecasting time series records. The LSTM architecture differs from the traditional perceptron structure because it incorporates a cellular and gates which controls the waft of statistics. Generally, the LSTM contains an input gate, a forget gate, an internal state (cell reminiscence), and an output gate as illustrated in the figure.

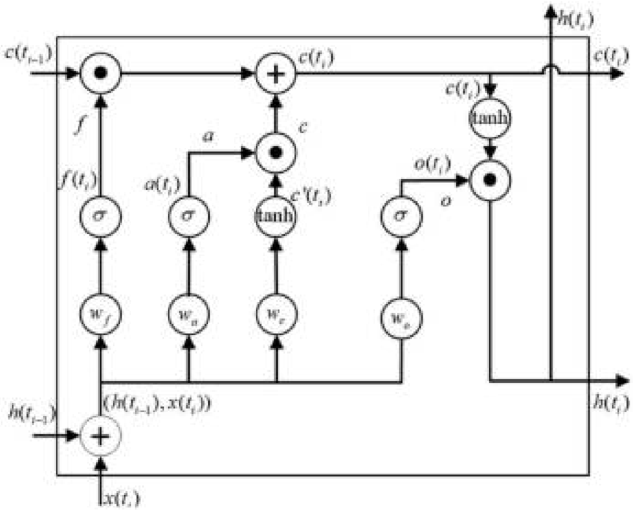

Figure 2.14: The LSTM's Structure
[1]

Where:

$x(t_i)$: The input value
$h(t_{i-1})$ and $h(t_i)$: The output value at time point t_{i-1} and t_i
$c(t_{i-1})$ and $c(t_i)$: Cell states at time points t_{i-1} and t_i
$b = \{b_a, b_f, b_c, b_o\}$ are biases of input gate, forget gate, internal state and output gate.
$\overrightarrow{W_1} = \{w_a, w_f, w_c, w_o\}$ are weight matrixes of input gate, forget gate, internal state and output gate.
$\overrightarrow{W_2} = \{w_{ha}, w_{hf}, w_{hc}, w_{ho}\}$ are the recurrent weights
$\overrightarrow{a} = \{a(t_i), f(t_i), c(t_i), o(t_i)\}$ are the output results for input gate, forget gate, internal state, and output gate.

Using these notations, the LSTM follow work is as follows: First, the forget f(ti) gate utilizes the two inputs: x(ti) and h(ti-1). Next, the information flows using sigmoid activation to c(ti-1) After that, the input gate a(ti) has taken the information it computes c(ti). Thereby, the output gate will use both sigmoid and tanh layers to adjust the result. Here is the mathematical expression for this explained process:

$$a(t_i) = \sigma(w_a x(t_i) + w_{ha} h(t_{i-1}) + b_a) \quad (3)$$

$$f(t_i) = \sigma(w_f x(t_i) + w_{hf} h(t_{i-1}) + b_f) \quad (4)$$

$$c(t_i) = f_t \times c(t_{i-1}) + a_t \times \tanh(w_c x(t_i) + w_{hc}(h(t_{i-1}) + b_c) \quad (5)$$

$$o(t_i) = \sigma(w_o x(t_i) + w_{ho} h(t_{i-1}) + b_o) \quad (6)$$

$$h(t_i) = o(t_i) \times \tanh(c(t_i)) \quad (7)$$

- Sigma and tanh: are activation functions
- x: Is a point-wise multiplication

Afterward, the learning process of the LSTM starts:

- Compute the LSTM output using Eqs. (1)–(5) (forward learning).
- Compute the error between the resulted data and input data of each layer.
- The error is reversely propagated to the input gate, cell, and forget gate.
- Based on the error term, the weight of each gate is updated using an optimization algorithm.

2.5.2.2 Hyperparameters for LSTM

An LSTM based learning framework contains various hyper-parameters:

- **The number of hidden layers:**[15] Typically, deep learning networks are used with a single-layer architecture. But, studies have shown that adding more layers would definitely lead us to an output roughly close to the desired output, hence a more accurate model. Therefore, defining the number of hidden layers is an important parameter.

- **The number of neurons:**[26] Each layer contains a certain number of neurons. Defining the right number for each neuron is a challenge. Indeed, if the number of neurons is lower than it should be (small number) this will impact the performance of the LSTM. In fact, the LSTM will not be able to memorize all the necessary information, thus affecting the predictions. On the other hand, if the number of neurons is high, an over-fitting problem will occur while the LSTM is in the training phase.
- **Learning rate:**[26] Applying a good learning rate is definitely a great addition that will help the model perform predictions optimally. Its function is to adjust the extent of the change to the weights of the model. It will be explained more properly in the next chapter.
- **Epoch size:**[26] One iteration corresponding to training the model for one time is one training epoch. The number of epochs should be chosen wisely because, with a small number, the model will not capture the patterns and with a high number, we risk overfitting the data.
- **Dropout rate:**[29] It's a booster to the output of the LSTM. It helps solve the overfitting problem.
- **Batch size:**[27] The batch size is an important hyperparameter which is the number of samples sent to the model at a single time. Generally, 64 is an optimal batch-size.

2.6 Conclusion

In this chapter we focused on the EDA part of a data scientist's job, which is exploring the data and treating it or preprocessing it so we can get it ready to the next part which it the modelling part.

Chapter 3

The Proposed Hybrid Deep Learning Model

3.1 Introduction

The main contribution of this research is the creation of a prediction model for anticipating detergent demand using advanced deep learning techniques. Convolutional layers, as discussed in the previous chapter, are good at extracting relevant knowledge and learning the internal representation of time-series data, whereas LSTM networks are good at detecting short-term and long-term dependencies. Our proposed model's main idea is to effectively combine the benefits of these deep learning techniques.
To this goal, we propose the CNN–LSTM model, which includes of two fundamental components: The first component includes of convolutional and pooling layers, which undertake complex mathematical operations to build features of the input data, while the second component uses the features obtained by LSTM, dense layers and other methods.

3.2 The CNN-LSTM Architecture

The architecture of the proposed CNN-LSTM based deep learning framework is shown in the figure below:

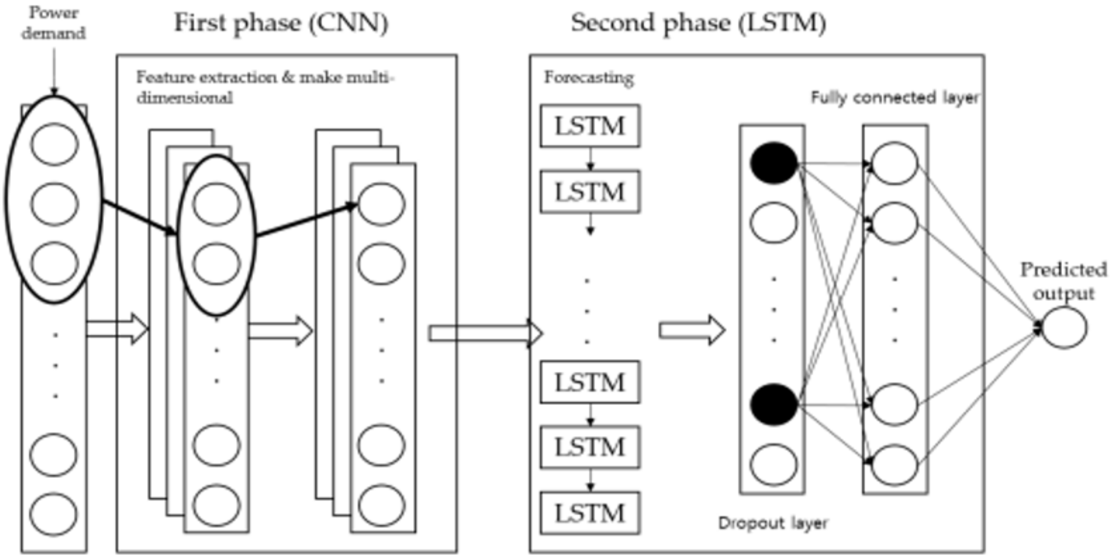

Figure 3.1: Hybrid Method structure
[38]

Convolutional operations are very effective, and stacking numerous convolutional layers in a deep learning framework allows the first layers to learn low-level features in the input. The output of the convolutional layers, the feature map, has the limitation of keeping track of the precise placement of the features in the input. It indicates that little changes in the input feature's location will result in a different feature map.
After the convolutional layer, a pooling layer is frequently added to mitigate the invariance limitation of the resulting feature map, while the activation function is used to improve the model's capacity to learn complicated structures.We've introduced a MaxPooling layer, which is a down-sampling approach that minimizes the spatial dimension of feature maps and hence reduces overall computing effort.

The dropout layer is an effective approach to avoid overfitting in the creation of any deep learning model. This layer involves the training process's random selection of neurons and inactivation of some of them.

The proposed model in our implementation consists of two convolutional layers of 32 and 64 filters of size (2), respectively, followed by a pooling layer, an LSTM layer, and a single neuron output layer, as shown in the figure.

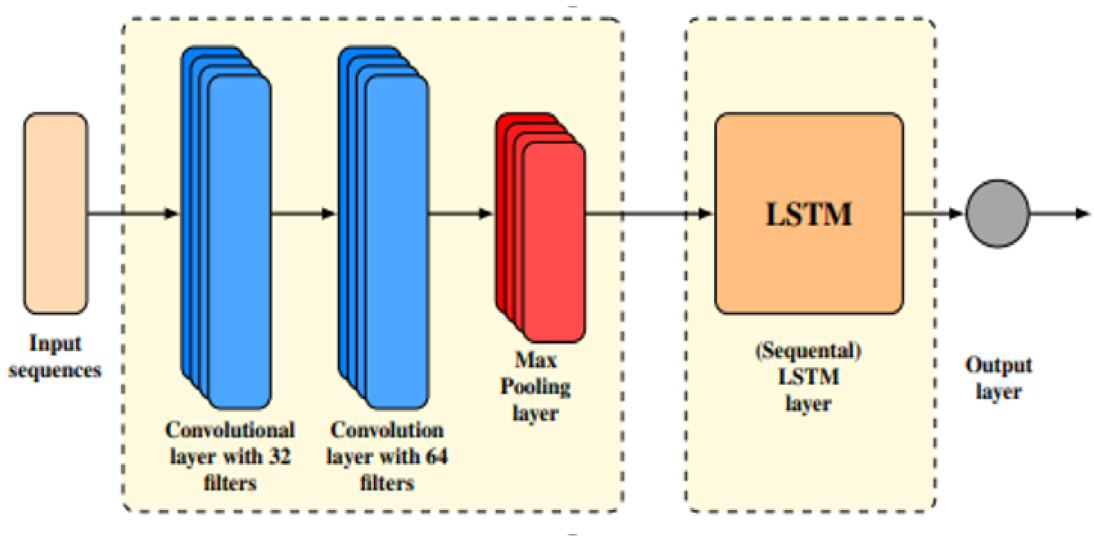

Figure 3.2: CNN-LSTM architecture with two convolutional layers, a pooling layer, an LSTM layer, and an output layer
[21]

- **The training flow of the proposed deep learning model:**
  The input data is divided into three categories: 70% training, 20% validation, and 10% test data. To track the validation loss, we used the mean absolute error (MAE) as a loss function. The training data and validation data are initially loaded, and the training procedure is started. The validation loss is calculated once each epoch is completed and checked to see if it is decreasing. The model is saved with the revised weights and the epochs are incremented if the validation loss is lowering. The learning rate is reduced and the epochs are increased if the validation loss does not decrease for ten consecutive epochs. When the epoch count hits 50, the training comes to an end.
  To avoid overfitting, we use the most recently saved best model for prediction and evaluation on the test data, as illustrated below:

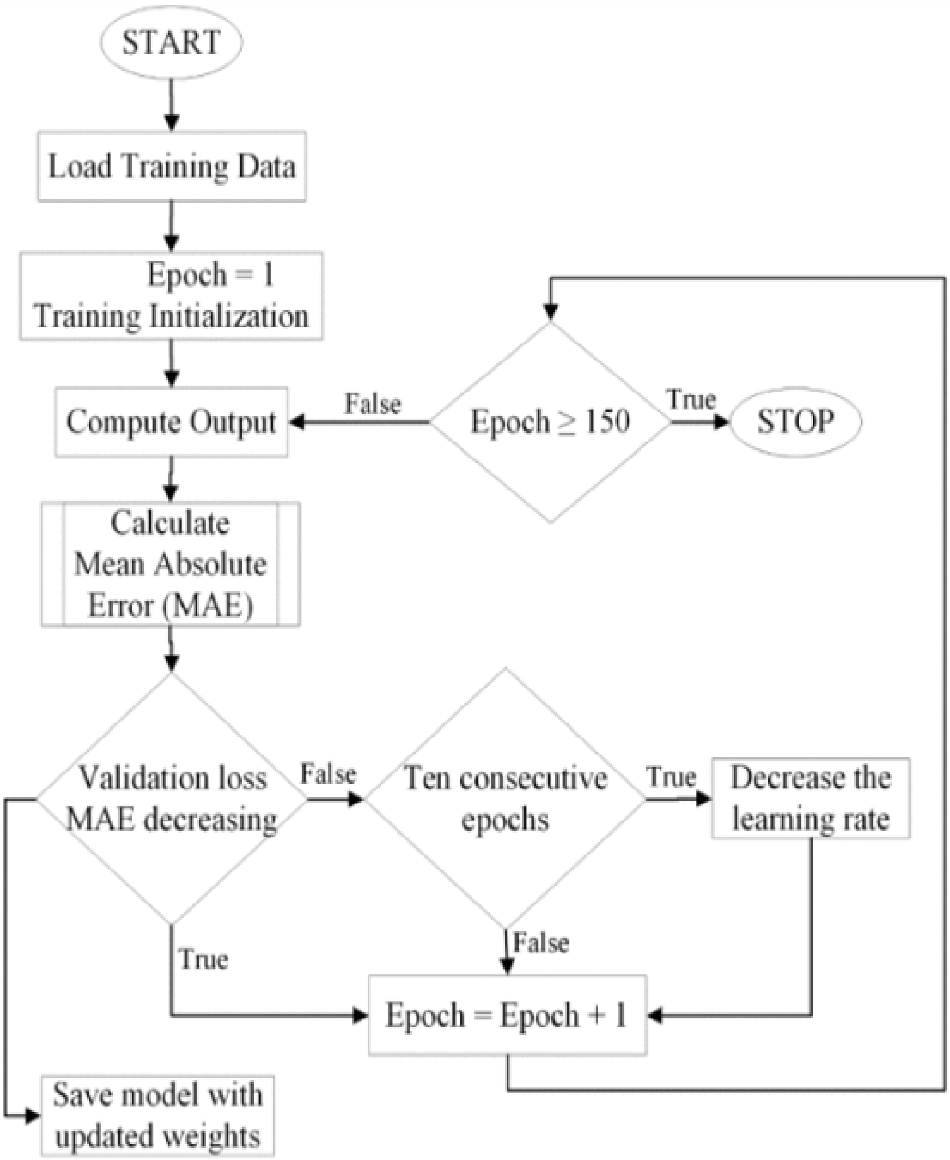

Figure 3.3: Hybrid Method structure
[22]

3.3 CNN-LSTM Training and Prediction Process

The CNN-LSTM model follows basically the following steps:

1. Gather the necessary data for predicting.

2. Data normalization (or data standardization). This is how it's done:

$$y_i = \frac{x_i - \overline{x}}{s}, \qquad (8)$$

$$x_i = y_i * s + \overline{x}, \qquad (9)$$

   Where yi stands for the standardized value
   xi: The input data
   S: The standard deviation of the input data
   X(bar): The average of input data

3. Initialization of the CNN-weights LSTM's and biases for each layer.

4. The CNN and pooling layers collect features from the input data.

5. The LSTM layer is used to calculate the CNN layer's output data.

Skipping, to **judgment state**:
To determine if the end condition is met,the end condition must have completed a particular number of cycles, the weights must be below a certain threshold, and the prediction error rate must be below a certain threshold. If any of the termination conditions are met, the training is finished, and the CNN-LSTM network is updated, so the next step is taken. Otherwise, return to the phase of error computation.

Error backpropagation:
Propagate calculated errors in the opposite direction, update the weights and deviations for each layer, and proceed to step 4 to continue training the network.

Prediction:
Input standardized data into the CNN-LSTM trained model to obtain the desired predicting.

Data standardized restore:
Because the CNN-LSTM model's output values were standardized, they are restored to their original values using the formula used in the second step.

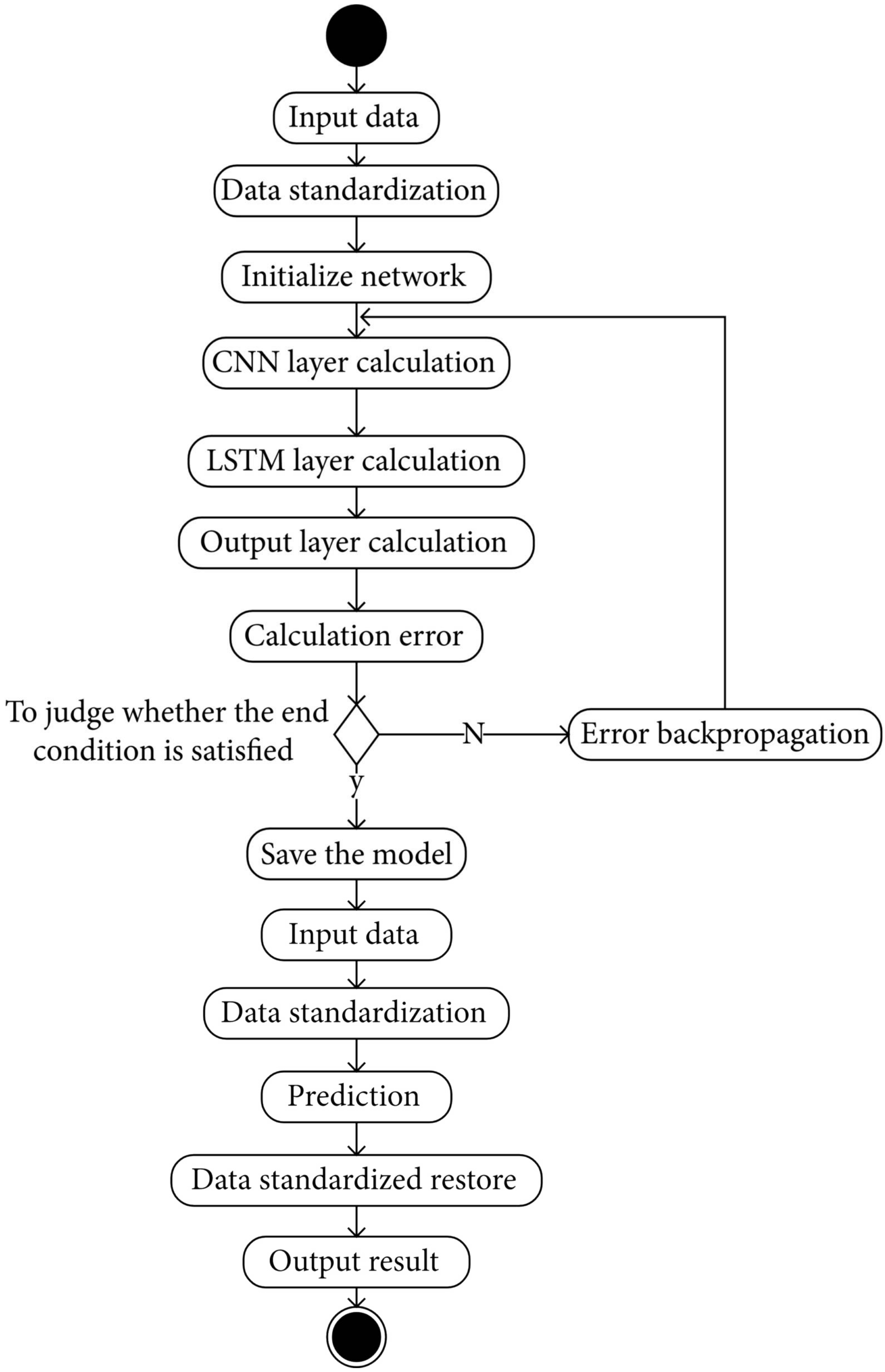

Figure 3.4: CNN-LSTM Training and Prediction Process [23]

3.4 Modelling

In this part we settled our first model's architecture and tuned our hyper parameters to better the loss and the validation loss of our model.
In our hyper parameters that we altered we have the loss function, the number of epochs, the windowed data set's parameters, the learning rate and the model's layers itself.
When it comes to the model's architecture, we started with the basic input and output layers and built our way up from there using the hybrid method. We tried some different architectures and fitted the model to the data and we archived the outputs of the last epochs every time. and then we chose the best in the end.
Our work in this part started with splitting our data to training set and validation set.
And then we went to the data preparation for the model. We windowed the dataset to windows that contains in the inputs three values of each product's sales numbers and in the outputs has one of each product's sales in the month after them.
At that point we have our data ready for the model.

```
[37] import tensorflow as tf
     def windowed_dataset(series, batch_size, n_past=24, n_future=24, shift=1):
         ds = tf.data.Dataset.from_tensor_slices(series)
         ds = ds.window(size=n_past + n_future, shift=shift, drop_remainder=True)
         ds = ds.flat_map(lambda w: w.batch(n_past + n_future))
         ds = ds.map(lambda w: (w[:n_past], w[n_past:]))
         return ds.batch(batch_size).prefetch(1)

     SPLIT_TIME = int(len(series) * 0.5)
     x_train = series[:SPLIT_TIME]
     x_valid = series[SPLIT_TIME:]

     tf.keras.backend.clear_session()
     tf.random.set_seed(42)

     BATCH_SIZE = 10
     N_PAST = 3

     N_FUTURE = 1

     SHIFT = 1
     train_set = windowed_dataset(series=x_train, batch_size=BATCH_SIZE,n_past=N_PAST, n_future=N_FUTURE,shift=SHIFT)
     valid_set = windowed_dataset(series=x_valid, batch_size=BATCH_SIZE,n_past=N_PAST, n_future=N_FUTURE,shift=SHIFT)
```

Figure 3.5: Windowing the dataset

The first line, from tensor slices, is one of the simplest ways to creates a dataset from a list in python.
The second line,ds.window, splits the dataset into small parts of size= nFuture + nPast with a shift of 1 time step when selecting the values to create the windows.

The Drop remainder = True means that all of the window will be of same length, which means when it gets to the end of the dataset it will stop at the last element that give us the window of the size that we specified.
The structure is now divided into batches of the primary size and the with the map function the elements are splitted into features and labels.
We did try few other variations like having the windows' inputs three months or six and the outputs 2 months but we saw that this approach gives us a better loss.

3.4.1 Model's Architecture

This is the architecture of the model that we found the best results with:

```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_7 (Conv1D)           (None, 3, 32)             224

 bidirectional_21 (Bidirecti  (None, 3, 64)            16640
 onal)

 bidirectional_22 (Bidirecti  (None, 3, 64)            24832
 onal)

 bidirectional_23 (Bidirecti  (None, 3, 32)            10368
 onal)

 dense_7 (Dense)             (None, 3, 6)              198

 lambda_1 (Lambda)           (None, 3, 6)              0

=================================================================
Total params: 52,262
Trainable params: 52,262
Non-trainable params: 0
```

Figure 3.6: Model's Architecture

in the modeling part, we did try some variations like adding more LSTM layers to the model and having them bidirectional so the model can follow the pattern of the data more, having the propagation of the information from the past to the future and from the future to the past works better than having it propagate only in one ways.
So in conclusion we found out that having three bidirectional LSTMs works best for the model.
To preserve the dimensionality we didn't add flatten layers and we noticed that adding a lambda layer at the end that multiplies by 400 improves the results

because the model can focus more on the small changes in the pattern and the inflate it in the last layer where we have a lambda function that multiplies the results by 400.

3.4.2 Tuning the Loss Function

In our work we noticed that the mean absolute error loss function is widely used in the time series' forecasting, so we trued it and found that it did optimise the mean absolute percentage error but didn't do a great job at getting us a good loss.
We did try the Huber loss function later on because it is widely used it time series' forecasting too and it is known to be more sensitive to out-layers in your data, that is why it can get a good loss. And that is exactly what we noticed after trying it.
After experimenting with these two loss functions we found out that a combination between the two of them can be made, and that this combination can give us the best results that we ever found.
We had to see how far we can get using this combination, so we had to test the number of epochs that can influence the improvement, which brings us to the next section.

3.4.3 Tuning the number of epochs

When choosing the epochs' number we have plotted the history of the loss for the first 200 and 150 epochs for the Mean Squared Error and the Huber Loss functions.

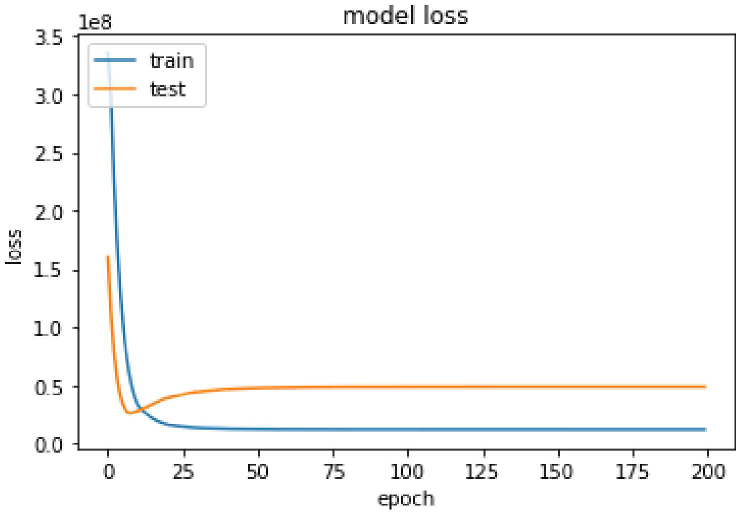

Figure 3.7: Epochs number for the mse
[1]

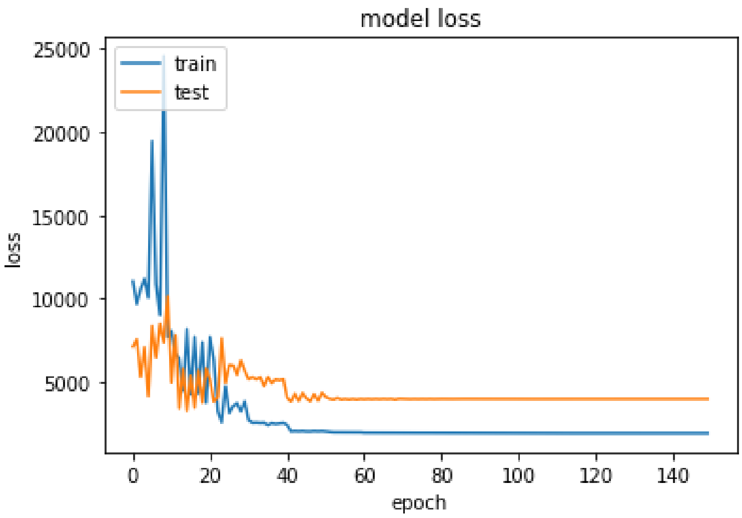

Figure 3.8: Epochs number for the Huber

After seeing these plots we know that our model won't improve any further if we increase the epochs' number more than 50 epochs in the training process when using the mean squared error loss function, and 50 epochs when using the Huber loss function.

3.4.4 Setting the learning rate

When setting up the learning rate, we found out that if we put it in a callback function that would decrease it as we move forward, would have good results. The learning rate of our optimizer should not be so high because that is how the model's weights diverge. This picture is a prefect example for how the learning rate, if too high, can make us go farther that where we want to go. And it shouldn't be too low either so that we can actually get to where we want to go in a good paste.

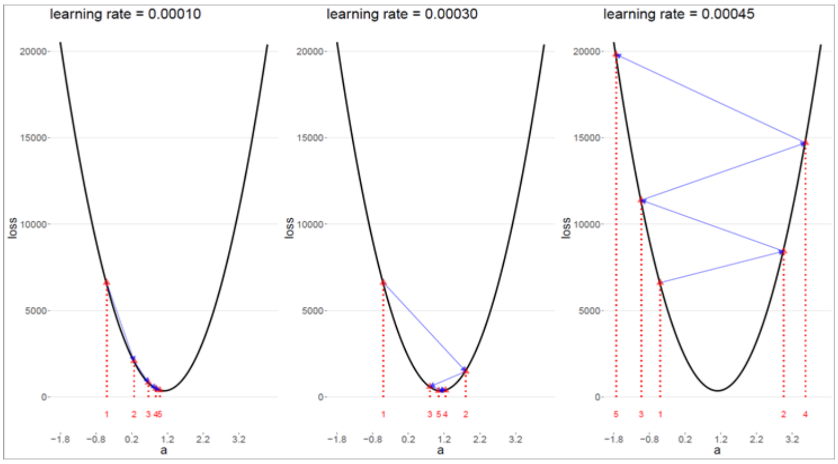

Figure 3.9: Accurate Learning Rate

```
[ ]  %matplotlib inline
     %config InlineBackend.figure_format = 'svg'

     def plot_lr(history):
         learning_rate = history.history['lr']
         epochs = range(1, len(learning_rate) + 1)
         plt.plot(epochs, learning_rate)
         plt.title('Learning rate')
         plt.xlabel('Epochs')
         plt.ylabel('Learning rate')
         plt.show()

     plot_lr(history)
```

Figure 3.10: Code for setting up the Learning Rate

To figure out the best learning rate that our model's optimizer needs for training,

we initiated the learning rate to a value of 0.01 and then the callback kept decreasing it's value after each epoch so that we get better and closer predictions after every epoch. When we get closer to the best loss, the improvement of the model seems to cease and freeze. The optimizer needs the learning rate that gives us better loss, that is why we need it to decrease as we go further with the training of our mode, but if it decreases too much the model won't be able to improve upon epochs
This is the plot for the evolution of our learning rate along the first training process.

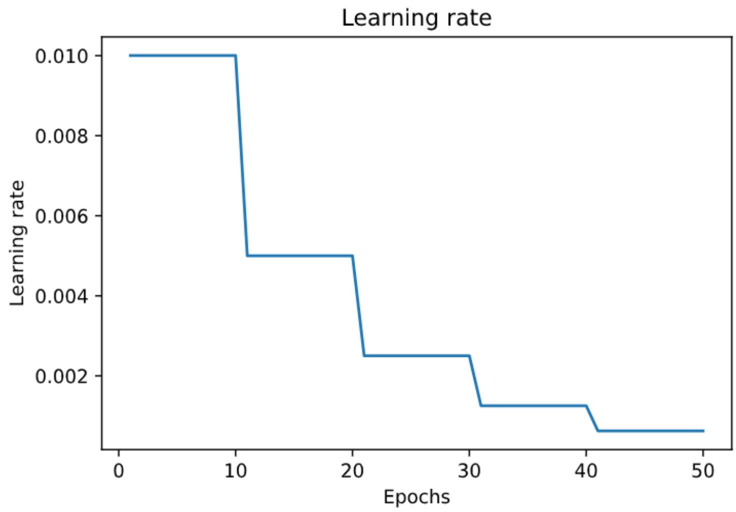

Figure 3.11: The Learning Rate Evolution

3.5 Fitting the Model

By now we have the model ready and the data windowed and ready to be fitted to our model.

3.5.1 Training the model

In our primary tests, we noticed that the when training the model with the mean squared error loss function, the mean absolute percentage error drops but the loss doesn't improve much. Yet when we train the model with the Huber loss function, the loss improves a lot yet the mean absolute percentage error doesn't. That is why we decided to combine the training methods to get the most out of the two of them.
At first we did fit the model with the mse loss function and the sgd optimizer and it went from 99.8319 mape to 38.6062 in 30 epochs.

After that we did fit the model with Huber loss function and the same optimizer having the learning rate shedualer as a callback. Here the loss dropped from 13386871.0 to 2654.4155.
After that we did the same process, running the model with the mse loss function and the sgd optimizer having the learning rate shedualer as a callback and the mae went from 2716.8787 to 2415.9082 in 10 epochs.
And then after running it with with Huber loss function and the same optimizer having the learning rate shedualer as a callback we reached 2311.6045 for the loss and a loss equal to 3097.1313 in 10 epochs.
In a final step we did the same process again and we noticed further improvement: We run the model with the mse loss function and the sgd optimizer having the learning rate shedualer as a callback and the mae reached 2777.5583 and a loss equal to 2777.0603 just in 30 epochs.
And then after running it with with Huber loss function and the same optimizer having the learning rate shedualer as a callback we reached a loss equal to 2774.5793 in 10 epochs.

3.6 Making Predictions

To make some predictions for the year 2021 we called the function predict and gave it a series of the past three months sales and it outputted three upcoming values for each product.
We know that for a Machine Learning model to give great predictions, we need a good amount of data, yet the model seems to give us good results that seem to follow the pattern. In our work we kept improving the model a lot along the process of making it.And we believe we went a long way, yet in the world of Machine Learning there always is room for improvement!

```
TP=[series[38],series[39],series[40]]
TP=np.expand_dims(np.array(TP), axis=0)
TP1=model1.predict(TP)
TP2=model1.predict(TP1)
TP3=model1.predict(TP2)
TP4=model1.predict(TP3)
```

```
print(TP1)
print(TP2)
print(TP3)
print(TP4)
```

```
[[[30682.928   26903.994   4111.6846   3232.1406   3732.1284   3187.7305]
  [32600.205   28670.365   4224.6484   3436.6924   3770.1633   3345.9006]
  [30918.74    27240.533   3936.5261   3307.7507   3468.1206   3189.0579]]]
[[[30682.928   26903.994   4111.6846   3232.1406   3732.1284   3187.7305]
  [32600.205   28670.365   4224.6484   3436.6924   3770.1633   3345.9006]
  [30918.74    27240.533   3936.5261   3307.7507   3468.1206   3189.0579]]]
[[[30682.928   26903.994   4111.6846   3232.1406   3732.1284   3187.7305]
  [32600.205   28670.365   4224.6484   3436.6924   3770.1633   3345.9006]
  [30918.74    27240.533   3936.5261   3307.7507   3468.1206   3189.0579]]]
[[[30682.928   26903.994   4111.6846   3232.1406   3732.1284   3187.7305]
  [32600.205   28670.365   4224.6484   3436.6924   3770.1633   3345.9006]
  [30918.74    27240.533   3936.5261   3307.7507   3468.1206   3189.0579]]]
```

Figure 3.12: Making Predictions

3.7 Comparing the loss with the ARIMA and LSTM models

After making the predictions with our model that was based on the hybrid method that combines Convollutional Neural Networks and Long Short-term Memory layers, we wanted to compare the accuracy of the forecast with with the other methods that our colleagues developed last year which are the ARIMA model and the LSTM model.

3.7.1 The results of all of the models

In the following figures are the results of the three models with the absolute errors of each prediction compared to the reality and the root mean squared errors between the sum of the four first months and the two after them, and the the average of all of the above among all the products.

Product		N-Reg-320	N-MS-320	D-MS-4000	D-MS-5000	D-Reg-4000	D-Reg-5000
February'21	SALES	26000	25500	3300	2600	3300	3100
	Hybrid	30683	26904	4112	3232	3732	3188
	Error	18%	6%	25%	24%	13%	3%
	Diff Squared	21930489,00	1971216,00	659344,00	399424,00	186624,00	7744,00
	ARIMA	18471	13282	2248	2635	3177	3840
	Error	29%	48%	32%	1%	4%	24%
	Diff Squared	56685841,00	149279524,00	1106704,00	1225,00	15129,00	547600,00
	LSTM	29325	14071	5029	1922	10531	2194
	Error	13%	45%	52%	26%	219%	29%
	Diff Squared	11055625,00	130622041,00	2989441,00	459887,42	52294013,73	821488,45
March'21	SALES	27000	27000	4000	2600	4300	3600
	Hybrid	32600	28670	4225	3437	3770	3346
	Error	17%	6%	5%	24%	14%	8%
	Diff Squared	31360000,00	2788900,00	50625,00	700569,00	280900,00	64516,00
	ARIMA	18817	16814	1353	1966	947	1842
	Error	30%	38%	66%	24%	78%	49%
	Diff Squared	66961489,00	103754596,00	7006609,00	401956,00	11242609,00	3090564,00
	LSTM	29114	23996	3629	1875	6852	6665
	Error	8%	11%	9%	28%	59%	85%
	Diff Squared	4468996,00	9024016,00	137641,00	525320,54	6513061,28	9395451,04
April'21	SALES	29000	29000	3800	2600	3800	3100
	Hybrid	30919	27241	3937	3308	3468	3189
	Error	7%	6%	4%	27%	9%	3%
	Diff Squared	3682561,00	3094081,00	18769,00	501264,00	110224,00	7921,00
	ARIMA	18913	14554	2591	2192	1854	1771
	Error	35%	50%	32%	16%	51%	43%
	Diff Squared	101747569,00	208686916,00	1461681,00	166464,00	3786916,00	1766241,00
	LSTM	29694	21474	3660	3971	4584	3081
	Error	2%	26%	4%	53%	21%	1%
	Diff Squared	481636,00	56640676,00	19600,00	1879037,81	614953,96	357,59

Table 3.1 : Comparison for February, March and April

May'21	SALES	26000	25000	3800	2600	4300	3100	
	Hybrid	30683	26904	4112	3232	3732	3188	
	Error	18%	8%	8%	24,3%	13%	3%	
	Diff Squared	21930489,00	3625216,00	97344,00	399424,00	322624,00	7744,00	
	ARIMA	19023	11365	1510	1017	2508	1834	
	Error	27%	55%	60%	61%	42%	41%	
	Diff Squared	48678529,00	185913225,00	5244100,00	2505889,00	3211264,00	1602756,00	
	LSTM	34305	26524	3425	2612	3138	5330	
	Error	32%	6%	10%	0%	27%	72%	
	Diff Squared	68973025,00	2322576,00	140625,00	149,33	1349825,71	4971517,50	
Average error	Hybrid	15%	6%	10%	25%	12%	4%	15%
	ARIMA	30%	47%	48%	26%	44%	39%	34%
	LSTM	14%	22%	19%	27%	82%	47%	21%
RMSE	Hybrid	4441,38	1694,06	454,45	707,23	474,44	148,26	2280,89
	ARIMA	8277,58	12724,33	1924,78	876,86	2136,35	1323,55	7292,92
	LSTM	4609,21	7046,44	906,55	846,23	3897,82	1948,64	4167,29

Table 3.2 : Comparison for May

June'21	SALES	29000	28100	3300	2600	3300	4100	
	Hybrid	32600	28670	4225	3437	3770	3346	
	Error	12%	2%	28%	32%	14%	18%	
	Diff Squared	12960000,00	324900,00	855625,00	700569,00	220900,00	568516,00	
	ARIMA	20273	18664	12546	894	12742	1270	
	Error	30%	34%	280%	66%	286%	69%	
	Diff Squared	76160529,00	89038096,00	85488516,00	2910436,00	89151364,00	8008900,00	
	LSTM	39767	28389	8243	3310	4250	5171	
	Error	37%	1%	150%	27%	29%	26%	
	Diff Squared	115928289,00	83521,00	24433249,00	504682,37	901873,11	1147576,56	
JuLY'21	SALES	26500	26500	3300	2100	3300	2100	
	Hybrid	30919	27241	3937	3308	3468	3189	
	Error	17%	3%	19%	58%	5%	52%	
	Diff Squared	19527561,00	549081,00	405769,00	1459264,00	28224,00	1185921,00	
	ARIMA	22822	21468	355	894	730	1806	
	Error	14%	19%	89%	57%	78%	14%	
	Diff Squared	13527684,00	25321024,00	8673025,00	1454436,00	6604900,00	86436,00	
	LSTM	36180	32315	4884	4599	9680	5331	
	Error	37%	22%	48%	119%	193%	154%	
	Diff Squared	93702400,00	33814225,00	2509056,00	6245250,90	40703634,40	10440653,44	
Average error	Hybrid	15%	5%	15%	32%	11%	14%	15%
	ARIMA	27%	40%	93%	38%	90%	40%	55%
	LSTM	37%	11%	99%	73%	111%	90%	70%
RMSE	Hybrid	4030,36	661,05	794,16	1039,19	352,93	936,60	1302,38
	ARIMA	6696,57	7561,72	6861,54	1477,31	6919,40	2011,88	5254,74
	LSTM	10237,94	4116,90	3670,31	1837,11	4561,00	2407,10	4471,73

Table 3.3 : Comparison for June and July

3.7.2 Discussion

When taking a first look to the second figure where we have the averages of the averages of the loss and the averages of the root mean square errors, we can conclude that the Hybrid method is giving much better results that the other two models. And we can confirm that if we go further in time to the following two months June and July. Our model seems to give more accurate predictions for most of the months and the products.
For some products we got a loss of 2 and 3 percent and the highest loss seems to be of 58 percent for our model, while the other models reached 286 percent of loss and surpassed the 100 and 200 percent several times!
we can see that our model is more efficient than ARIMA and LSTM models. In fact, like all neural networks, it does not impose any hypothesis on the distribution of the data and it is able to learn linear and non-linear relationships and to model complex interactions. And the addition of the CNNs did improve the efficiency of the LSTMs. So our model seems to be more stable and reliable.

3.8 Conclusion

As a conclusion for this chapter, here is a general overview for our work: In our pipeline we started with connecting the environment to the drive and then accessing to the data, the commas separated values file, and then we did some data pre-processing. After that we Plotted some figures and figured out that the products sales are on different scales, that is why we normalized the data so we can make more accurate conclusions. And then we went to the data preparation for the model where we windowed it. We then prepared the model and trained it on the data. After that we made some parameter tuning where we changed the parameters of the model, added some layers to better the performance. At last we tried to predict some values for the upcoming two months.

General Conclusion

In industrial complexes, sales and demand forecasts play a significant role in the proper formulation of strategic, tactical, and operational choices. In demand forecasting, data mining techniques such as time series and artificial neural networks have been frequently used.

To begin, time-series approaches and artificial neural networks were used to forecast the sales of the six Henkel detergent products four from the Dixan category and two from the Nadhif category.

There is a wide range of products available, each with its own set of trends. It's critical to select the best method for each product. To reduce forecasting inaccuracy, the most suited approach among time series and the artificial neural network was chosen to forecast each product's future demand.

As a result, we went over data Pre-processing and looked through the data we were given in order to deploy this ANN approach. Then, we got to the interesting part: modeling and receiving predictions, comparing them to real values, and comparing them to forecasts given by other models constructed by our colleagues using ARIMA and LSTM last year.

For the modeling part, the CNN is used to extract the input data's features. LSTM is used to learn the extracted feature data and forecast the products' closing demand in the coming months. As mentioned earlier the appropriate data from Henkel's is used in this report to corroborate the experimental results.

As a conclusion, in comparison to the LSTM and ARIMA, the CNN-LSTM has the highest forecasting accuracy and best performance, while MAE is the smallest. CNN-LSTM is a good choice for forecasting product demand. CNN-LSTM additionally proposes practical experience for persons conducting financial time series data investigation. However, there are certain flaws in the model.It solely evaluates the impact of product demand data on closing sales, for example, and ignores emotional aspects like news, crisis periods and national policy in the prediction. Our future study will primarily focus on improving sentiment analysis of demand-related news and national policies in order to ensure demand forecast accuracy.

Bibliography

[1] *Abbasimehr, Hossein, Mostafa Shabani, and Mohsen Yousefi. "An optimized model using LSTM network for demand forecasting." Computers industrial engineering 143 (2020): 106435.*

[2] *Armstrong, J. Scott and Fred Collopy. 1998. "Integration of statistical methods and judgment for time series forecasting: Principles from empirical research." In Forecasting with Judgment. Eds. George Wright and Paul Goodwin. Chichester: John Wiley.*

[3] *Armstrong, J. Scott. "Judgmental bootstrapping: Inferring experts' rules for forecasting." In Principles of forecasting, pp. 171-192. Springer, Boston, MA, 2001.*

[4] *Armstrong, Jon Scott, and Kesten C. Green. Demand forecasting: evidence-based methods. No. 24/05. Monash University, Department of Econometrics and Business Statistics, 2005.*

[5] *Brown, S., Blackmon, K., Cousins, P., and Maylor, H. (2013). Operations management: policy, practice and performance improvement. Routledge.*

[6] *CARROD Bryan and FYALL Alan. "Revisiting Deplhi": Tourism Research Methods: Integrating Theory with Practice. 2005.*

[7] *Chopra, S., and Meindl, P. "Supply chain management: Strategy, planning, and operation (6th ed.)", Upper Saddle River, New Jersey: Pearson Education, Inc. (2016).*

[8] *Claycamp, Henry J. and Lucien E. Liddy. 1969. "Prediction of new product performance: An analytical approach." Journal of Marketing Research 6: 414-420.*

[9] *Da Costa Lewis, N. "Neural Networks for Time Series Forecasting with R". (2017).*

[10] *Des Travaux Publics, École Hassania. "PREVISION DE LA DEMANDE DANS LA CHAINE LOGISTIQUE COMBINAISON D'UN MODELE CAUSAL AVEC LES ALGORITHMES GENETIQUES.".*

[11] *Funahashi, K.I. "On the approximate realization of continuous mappings by neural networks. Neural networks", 2(3), pp.183-192, (1989).*

[12] *Gordon, Theodore Jay. "The delphi method." Futures research methodology 2, no. 3 (1994): 1-30.*

[13] *Green, K. C., Armstrong, J. S., Graefe, A. 2007. Methods to Elicit Forecasts from Groups: Delphi and Prediction Markets Compared. Foresight, 8, 17-20. Available from http://kestencgreen.com/greenarmstrong-graefe-2007x.pdf.*

[14] *Hansun, S. (2013, November). A new approach of moving average method in time series analysis. In 2013 Conference on New Media Studies (CoNMedia) (pp. 1-4). IEEE.*

[15] *Hermans, M., Schrauwen, B. (2013). Training and analysing deep recurrent neural networks. Paper presented at the Advances in neural information processing systems.*

[16] *Hornik, K.; Stinchcombe, M.; White, H. "Multilayer feedforward networks are universal approximators". Neural networks, 2(5), pp.359-366,(1989).*

[17] *Hyndman, R. J. (2011). Moving averages. International encyclopedia of statistical science, 866-869.*

[18] *Hyndman, R. J. (2011). Moving averages. International encyclopedia of statistical science, 866-869.*

[19] *Hyndman, R. J., and Athanasopoulos, G. (2018). Forecasting: principles and practice.OTexts.*

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

[21] *Kim, Myoungsoo, Wonik Choi, Youngjun Jeon, and Ling Liu. "A hybrid neural network model for power demand forecasting." Energies 12, no. 5 (2019): 931.*

[22] *Livieris, Ioannis E., Emmanuel Pintelas, and Panagiotis Pintelas. "A CNN–LSTM model for gold price time-series forecasting." Neural computing and applications 32, no. 23 (2020): 17351-17360.*

[23] *Lu, Wenjie, Jiazheng Li, Yifan Li, Aijun Sun, and Jingyang Wang. "A CNN-LSTM-based model to forecast stock prices." Complexity 2020 (2020).*

[24] *Meade, Nigel and Towhidul Islam. 2001. "Forecasting the diffusion of innovations: Implications for time series extrapolation." In Principles of Forecasting. Ed. J. Scott Armstrong. Norwell, MA: Kluwer Academic Publishers, 577-595.*

[25] *Müller, A. C., and Guido, S. (2016). Introduction to machine learning with Python: a guide for data scientists. " O'Reilly Media, Inc.".*

[26] *Reimers, N., Gurevych, I. (2017). Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. arXiv preprint arXiv:1707.06799.*

[27] *Shi, H., Hu, S., Zhang, J. (2019). LSTM based prediction algorithm and abnormal change detection for temperature in aerospace gyroscope shell. International Journal of Intelligent Computing and Cybernetics, 12(2), 274–291.*

[28] *Singh, Arunesh Kumar, S. Khatoon Ibraheem, Md Muazzam, and D. K. Chaturvedi. "An overview of electricity demand forecasting techniques." Network and Complex Systems 3, no. 3 (2013): 38-48.*

[29] *Srivastava, N., Hinton, G. Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929–1958.*

[30] *Stoll, Franz. "A Comparison of Machine Learning and Traditional Demand Forecasting Methods." (2020).*

[31] *Tull, Donald. S. 1967. "The relationship of actual and predicted sales and profits in new product introductions." Journal of Business 40: 233-250.*

[32] *Van Nguyen, Truong, Li Zhou, Alain Yee Loong Chong, Boying Li, and Xiaodie Pu. "Predicting customer demand for remanufactured products: A data-mining approach." European Journal of Operational Research 281, no. 3 (2020): 543-558.*

[33] *Weiss, G., and Davison, B. (2010). Data Mining, Handbook of Technology Management.*

[34] *Williams, T. M. "Adaptive Holt-Winters forecasting", Journal of the Operational Research Society, 553-560, (1987).*

[35] *WZhang, X. "Time series analysis and prediction by neural networks", Optimization Methods and Software, 4(2), pp.151-170, (1998).*

[36] *Yu, Y.; Choi, T.-M.; Hui, C.-L. "An intelligent fast sales forecasing model for fashion products", Experts systems with application, 38, pp. 7373-7379, (2011).*

[37] *Yu, Yong, Tsan-Ming Choi, and Chi-Leung Hui. "An intelligent fast sales forecasting model for fashion products." Expert Systems with Applications 38, no. 6 (2011): 7373-7379.*

[38] *Yu, Yong, Tsan-Ming Choi, and Chi-Leung Hui. "An intelligent fast sales forecasting model for fashion products." Expert Systems with Applications 38, no. 6 (2011): 7373-7379.*

[39] *Zhang, G.; Patuwo, B. E.; Hu, M. Y. "Forecasting with artificial neural networks: The state of the art", International journal of forecasting, 14(1), pp.35-62,(1998).*