**Cairo University**

**Faculty of Computers & Artificial Intelligence**

**Operations Research & Decision Support Dept.**

# Retail Sales Forecasting

The Graduation Project Submitted to
The Faculty of Computers and Artificial Intelligence,
Cairo University
In Partial Fulfillment of the Requirements
for the bachelor's degree

In

**Operations Research and Decision Support**

*Under Supervision of:*
**Dr/Doaa Saleh**

**CAIRO UNIVERSITY**

**July/2023**

# Retail Sales Forecasting

**Submitted by:**

**Sara Adel Basha 20190232**

**Mohamed Rizk Abdelfattah 20190439**

**Mohamed Eslam Amin 20190419**

*Supervised By*

**Dr/Doaa Saleh**

**CAIRO UNIVERSITY**

**Feb 2023**

# ABSTRACT

Formulation of the tasks by the managers contributes to planning and taking appropriate actions and allows for evaluation of the enterprise's performance. Taking right decisions is preceded by an in-depth analysis of the resources and means available in the enterprise and the tools and methods that can be used. Therefore, it is essential that forecasting is used in decision-making processes as it might contribute to improved accuracy of decision-making. Forecasting in production enterprises allow for finding the most probable course of processes. When defining the most important tasks for the enterprise, the managers should be based on the forecasts for :

- demand for the goods that can be manufactured by the enterprise,
- demand in individual market segments,
- general economic conditions,
- technological changes,
- actions taken by competitors,
- industry.

The aim of the forecasting system in any enterprise is to provide information about future changes in the business environment and the impact of these changes on the enterprise in the form of forecasts. Using the tasks of this system might be assigned to a special department or to current units that use the forecasts prepared. Therefore, the study will be focused on data analysis, Machine learning, and interface tools.

.

# PLAIGRISM CERTIFICATE

This is to certify that the project entitled "**Retail Sales Forecasting**", which is being submitted here with for the award of the "**Bachelor of Computer and Artificial Intelligence Degree" in "Operations Research and Decision Support"**. This is the result of the original work by Sara Adel, Mohamed Rizk, and Mohamed Eslam under my supervision and guidance. The work embodied in this project has not been done earlier for the basis of award of any degree or compatible certificate or similar tile of this for any other diploma/examining body or university to the best of my knowledge and belief.

**Turning Originality Report**

Processed on 28-feb-2023.

ID: 300502964

Word Count: 12948

Similarity Index

10%

Similarity by Source

| | |
|---|---|
| Internet Sources: | 06% |
| Publications: | 0 % |
| Student Papers: | 08% |

Date: 28/02/2023

Supervisor Name Dr/ Doaa Saleh

# DECLARATION

We hereby declare that our dissertation is entirely our work and genuine / original. We understand that in case of discovery of any PLAGIARISM at any stage, our group will be assigned an F (FAIL) grade and it may result in withdrawal of our bachelor's degree.

Group members:

| **Name** | **Signature** |
| --- | --- |
| Sara Adel Basha | ____________________ |
| Mohamed Eslam Amin | ____________________ |
| Mohamed Rizk Abdelfattah | ____________________ |

# ACKNOWLEDGMENT

We offer all appreciation and respect to Dr. Doaa Saleh for her efforts with us and ensuring that she is with us in any assistance and any problem. And had it not been for her opinion on our work, we would not have reached this stage of the project.

# TABLE OF Content

**Chapter** **Page**

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1

## 1.0 Introduction

"Retail sales" refers to an economic metric that tracks consumer demand for finished goods. So, retail sales are seen as a stand-in for consumer spending, and by extension, can be seen as a key measure of the health of the economy. In view of its importance, we must look at the problems that most retailers face, the most important of which is the problem of the time of shipping products to meet the needs of customers, and at the same time the problem of the cost of storing products because of not knowing what customers demand, so most retailers have to many required products, which may not be needed.

## 1.1 Problem Domain

The best solution to the retailers' problem is to anticipate or know the required products during the coming period to save shipping time and storage costs, increase profit for the presence of products, and win the confidence of customers and their love for the store. Maintain a successful business. This is because most critical decisions are bound to be based on these expectations. We need to identify models that can be applied to solve the problem because they cannot be based on guesswork. These models can help retailers estimate the quantities of products they need to stock and the timing of their orders. Techniques such as time series analysis, regression analysis, and machine learning algorithms can be applied to develop accurate demand forecasting models.

## 1.2 Problem Statement

Have you ever walked into a retail store and had the thrill of making a purchase on a product you discovered, only to find that it was out of stock? This can be a frustrating experience for customers and is very common. Product unavailability is a common complaint in the retail industry, so retailers are under market pressure to meet customer demand. Retailers do not always know how much to put in stock to meet customer requirements. Some of the problems that can occur are that retailers keep inventory levels low, and in this way increase the risk of being out of stock. On the other hand, having very high inventory levels will increase the cost related to inventory handling. Companies seeking to achieve good sales performance often need to maintain a balance between meeting customer demand and controlling inventory costs.

The motivation for each is different as companies balance the problem of having too much inventory (which can lead to high costs of overstocking, leading to issues such as restricted capital, inventory in writing but enabling customer demand to be consistently met) against having too little inventory Very often (which can lead to lost sales, lower customer satisfaction, and other issues). So, companies must be alerted to changing conditions that may require more careful inventory management.

## 1.3 Proposed system

FMCG retailer, also known as a fast-moving consumer goods retailer, is a business that specializes in selling fast-moving consumer goods to end consumers. These products are non-durable goods that are typically low-cost, high-volume items that are purchased frequently and consumed rapidly. These products are a staple in households and include items such as food and beverages, personal care products, cleaning supplies, and over-the-counter medications. These retailers play a crucial role in the distribution and sale of FMCG products, ensuring that these goods are readily available to customers in various locations. So, the proposed system is used to predict sales using different models. In the proposed system, we will analyze the sales data of one of FMCG retailer specializing in cleaning supplies is a retail business and will predict the sales for the coming weeks and for a specific category. For this kind of project of sales prediction, we will evaluate the result based on the training and testing set of the data.

### 1.3.0 Aims and Objectives:

The objective of this project is to estimate precisely the category sales forecasting for detergents. In Order to Perform Predictions on Various categories of detergents, the results of the manual forecasting are not that great and varied across stores. There are many factors in the dataset that might impact sales, for example, Quantity and the price of unit. So, prediction models have been implemented to increase the precision. Sales managers depend greatly on the information provided by accurate sales forecasting to guide their business decisions. In order to maximize sales and revenue while delegating resources and sales reps more efficiently, sales managers need to be able to predict the future performance of their organization as closely as possible.

We aim to manage the business effectively, provide an outlook into the near-term predictable future of the business, facilitate business planning, risk management and help with budgeting and goal setting for the company, identify potential early warning signals and risks in the sales pipeline to help sales teams achieve their goals and to give a chance to tackle these issues at an early stage before they affect the performance of the sales team. With these warning signals and risks mitigated, reps can more effectively work on the right prioritized opportunities. Preprocessing will also be done on the dataset before modeling, including handling the outliers and missing values and performing spectral analysis. The trends and patterns across the data will be investigated.

### 1.3.1 Proposed system features

We can generate future insights from the results of forecasting models with a significant degree of precision, any organization can now use past and current data to reliably forecast sales milliseconds, days, or years into the future. We can use the results of forecasting models to optimize supply chains. This means that at the time of order, the product will be more likely to be in stock, and unsold goods won't occupy prime retail space. In our forecasting tool, we can build new features from existing ones to achieve higher forecast accuracy.

## 1.4 Development Methodology

In this study, we built a different model as Autoregressive Moving Average (ARMA) Model, Multi-layer Perceptron (MLP) neural network model which is a clans of feedforward artificial neural network (ANN) and Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture.

The input-output datasets were first normalized. The rescaling (min-max normalization) is the normalization technique used to scale the values in the range [0, 1]. This transform features by scaling each feature to a given range.

The input and output datasets are both split into 30 percent for testing and 70 percent for training.

Validation and Testing: Validate the forecasting models using out-of-sample data or cross-validation techniques to assess their performance on unseen data. Perform sensitivity analysis and assess the models' robustness to different scenarios and potential changes in market conditions.

## 1.5 Resource Requirement

The analysis for this study has been performed using some main tools: Python, and Excel.

Python because it provides an easy experience for ML that no other language quite can. The models and Exploratory Data Analysis have been executed using development tools like Anaconda, google Collab, and PyCharm.

Study focuses on:

Data Analysis: Good data and understanding for how to make analysis for data are the most important requirements for a good sale forecast.

Machine learning: Basic understanding of ML and its real-world applications is an important requirement for a good sales forecast.

# CHAPTER 2

## 2.0 Introduction

FMCG store for detergents is an Egyptian retail business that provides soap, shower Gel, among other products. The challenge for a retail industry is to ensure that warehouse and supply chain space utilization is optimized to ensure supply effectively meets demand. This is where accurate sales forecasting enables stores to see insights into the future period and make business decisions. Popular approaches to forecasting problems.

## 2.1 Overview of project

We collect data from FMCG store for detergents, which is an Egyptian retail store, the dataset was used for building a sales forecasting models to generate future insights from the results forecasting model with a significant degree of precision, any organization can now use past and current data to reliably forecast sales milliseconds, days, or years into the future.

By using the sales forecasting approaches, obtained sales forecast results could help the organization to assess the sales of the goods, to maintain stock of the goods which have more sales, and saves money and time of planning and management of the items which have the least sales or no sales in a particular region.

## 2.2 Limitations of project

The study has been conducted on sales data that belonged to the Store sales and it cannot be assured that similar results will be obtained from the study conducted on the sales data belonging to the other region as the sales may vary in other regions. Due to the unavailability of the company's information like customer details, certain campaigns and discounts. They haven't been included in the data which would benefit in obtaining better forecasts.

## 2.2.0 Innovations in project

Innovation is important to meet the scalability of any project. In the retail world, most of the stores' structure depends mainly on 5 elements: executive, sales, marketing, finance, and budget. Due to low budget or lack of experience, they are usually missing the IT or data science which result in exerting too much effort and wasting much time which can be highly they used any of new technological trends especially those in the Al and data science in this project, we are building a forecasting tool to support retail stores in decision instead of asking for freelancers' help or being forced to manually calculating sales process. The main target is to save time, effort, and money for the managers using this tool.

Our forecasting tool is mainly a neural network model which analyses the sales data with all the sales conditions. For the sake of simple and easy usability, we are providing these insights in the form of visuals that visualize all aspects of the business, which is a fashion store in our case, in a very simple way. It also provides key indicators in illustrated points to over facilitate interpretation of the visuals. Moreover, after presenting the insights, it also provides foresight. In other words, it provides prediction for the sales profit of each category for a specific time period to optimize the buying and selling processes and the production process. The interface of the tool makes it so easy for anyone with even zero programming skills to get the best use of it with just a click. Although there have been existing research focusing on sales forecasting by using statistical techniques in the automotive industry, there has been no research based on developing sales forecasting related to the stores based on machine learning, which can be used for the sales team of any wide range of companies and organizations in the automotive industry. This thesis suggests that the machine learning approach can be used to get insights from the data and to forecast the sales of stores. This approach can be further used to develop advanced forecasting tools which can be able to forecast more precise forecasts.

### 2.2.1 Design of project

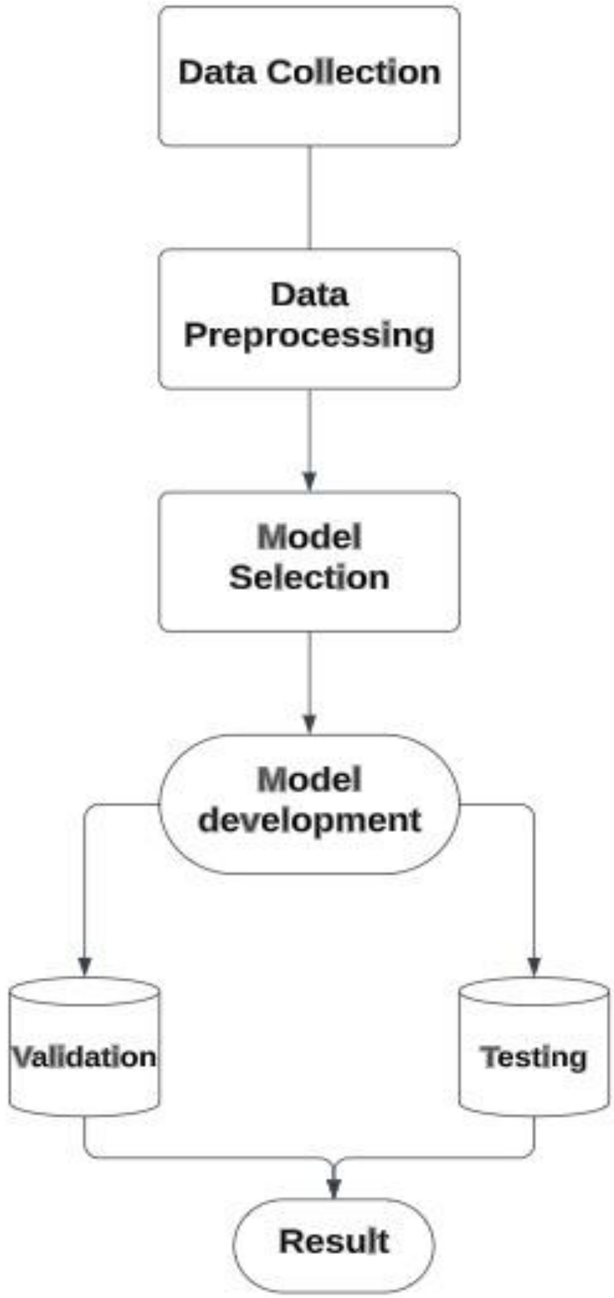


Fig 2.1 design of project

# CHAPTER 3

## 3.0 Data Cleaning /Understanding

Before starting the implementation, we had to make sure that our data is ready for use through some data analysis processes, which was a major factor in understanding it and understanding the relationships in it. Therefore, we had to prepare the data by:

- ✓ Collect → Cleaning → Exploring → Model.

We can make sure that the data is ready for use by eliminating the following problems:

- ✓ Duplicate data -- outdated data -- Incomplete data -- Incorrect data -- Inconsistent data

<u>Description of data:</u>

| | |
|---|---|
| Date | The date on which the order has made. |
| Product ID | Each product has an ID representing it. |
| Category | We have mainly 3 categories of the products |
| Sub-category | Each main category has many sub-categories. |
| Units | The number of pieces requested in one order from the store. |
| Price | The price of each units |
| Total Sales | Total sales from units. |

Tab1 Description of data

Dataset have 3 main categories of products we can describe them in this table:

| Category | Number of Records |
|---|---|
| Soap | 125595 |
| Shower Gel | 500 |
| Detergent | 3334 |

Tab2 Main Category

## 3.1 Data Exploring Results /insights.

Exploratory data analysis is performed after cleaning and understanding the data.

### Total Sales by Category

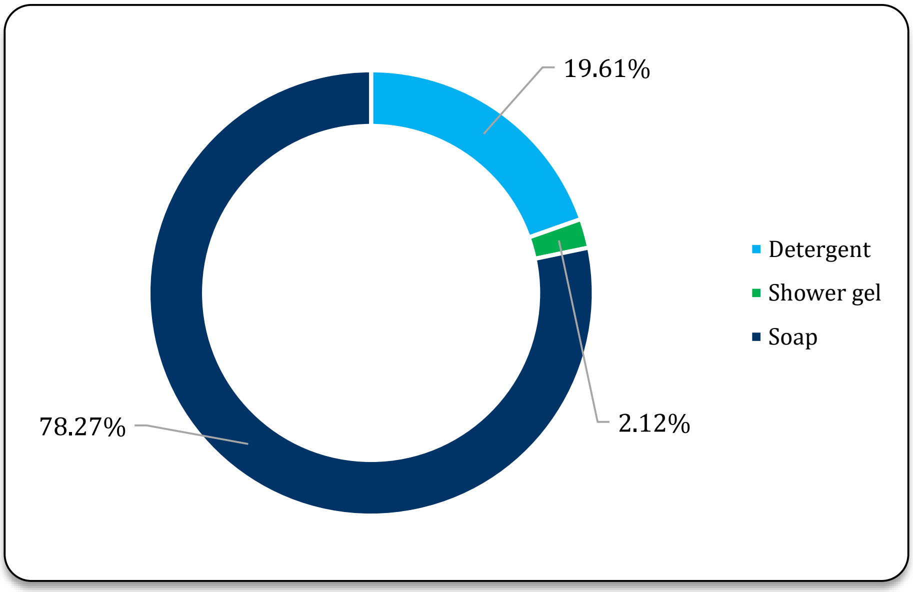


Fig 3.1 Total Sales by Category

### Total Sales by Subcategory

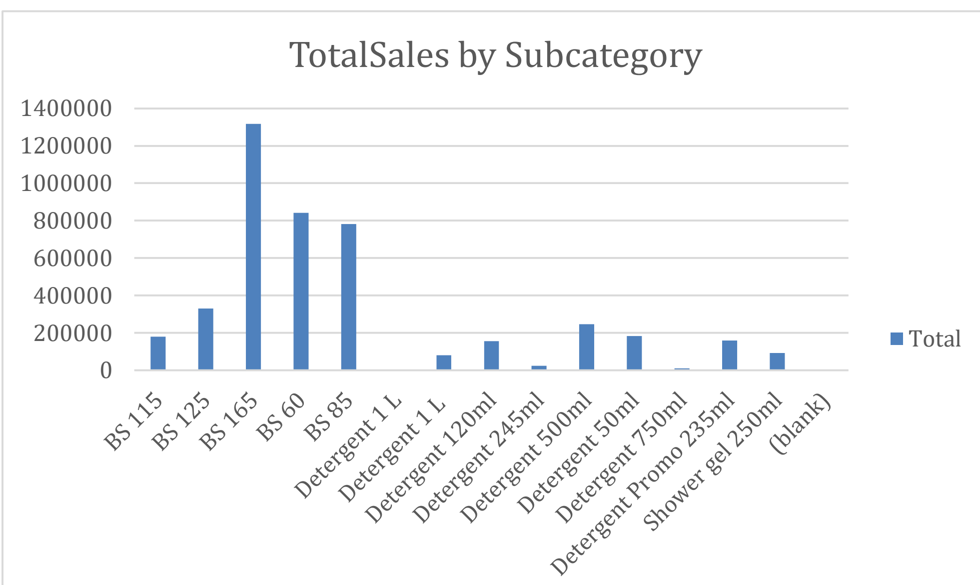


Fig 3.2 Total Sales by Subcategory

## Monthly Total Sales

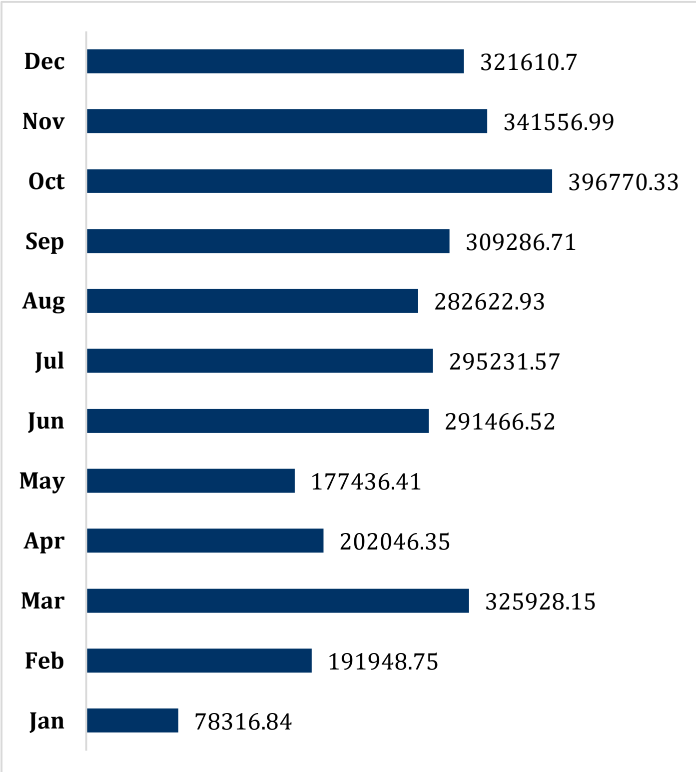


Fig 3.3 Monthly Total Sales

# 3.2 Data Manipulation

After cleaning and exploring, it is time to manipulate the data to obtain the most value out of it. This is done by combining the data and including all the different sources to narrow the range of private data. We all have the main data inputs which are product categories. We will summarize the entire data based on the weekly or monthly data and calculate the average sales profit for each category in each week or each month.

Where the difference between each time period is almost constant, replacing it with fixed numbers starting from zero and calculating the correlation between all the inputs to take it as a basis in the model stage. The data is divided for implementation by taking part of it for training and testing to reach a more accurate prediction by calculating the error.

# CHAPTER 4

## 4.0 Time Series Analysis

Time series analysis is a statistical method used to analyze and interpret data points collected over time. It focuses on understanding the patterns, trends, and relationships within the data to make predictions and forecasts about future values. Time series data typically consists of observations taken at regular intervals, such as hourly, daily, monthly, or yearly.

Key Concepts in Time Series Analysis:

Time Series Components: A time series can be decomposed into four main components:

•Trend: The long-term pattern or direction of the data.

•Seasonality: Repeating patterns or variations that occur at fixed intervals.

•Cyclical: Patterns that occur over extended periods but do not have fixed intervals.

•Random or Residual: Unpredictable and irregular fluctuations not accounted for by the other components.

During the analysis stage, one of the key objectives is to identify and analyze the underlying trend present within the dataset.

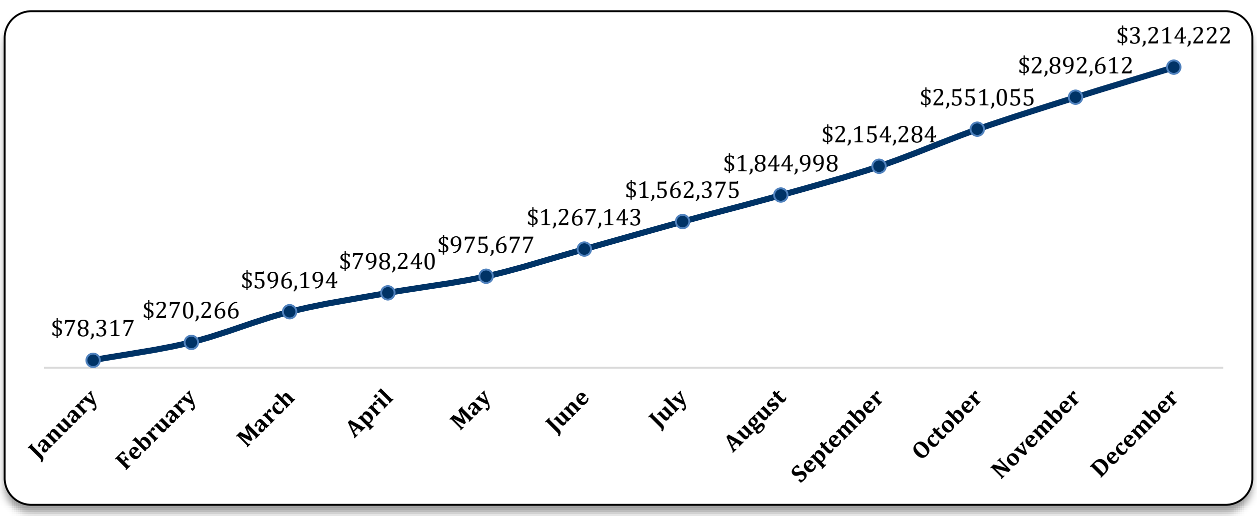


Fig 4.0 Running Total Sales

## 4.1 ARIMA

The ARIMA (Autoregressive Integrated Moving Average) model is a widely used statistical model for analyzing and forecasting time series data.

## 4.1.0 Components of ARIMA Model:

The ARIMA model consists of three main components:

**a)** Autoregressive (AR) Component: It models the relationship between an observation and a linear combination of its past values. The AR component captures the influence of the lagged values on the current value.

**b)** Moving Average (MA) Component: It models the dependency between an observation and a residual error from a moving average process. The MA component captures the influence of the past errors on the current value.

**c)** Integrated (I) Component: It incorporates differencing to transform a non-stationary time series into a stationary one. Differencing removes trends and seasonality by subtracting the previous value from the current value.

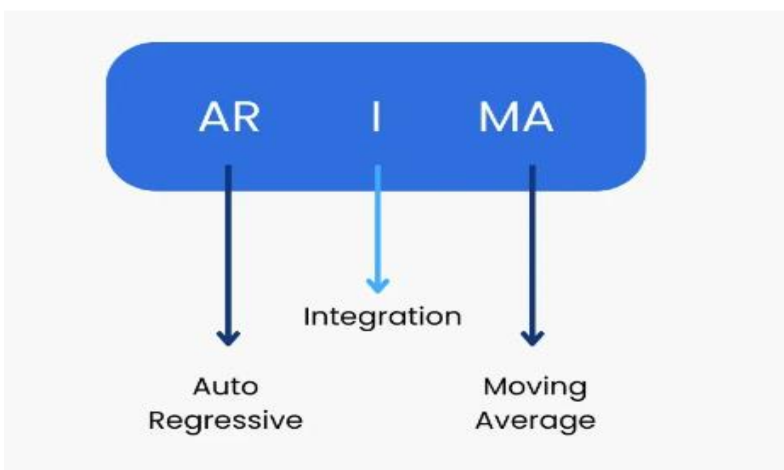


Fig 4.1.0 Component of ARIMA

## 4.1.1 Stationarity in Time Series

Stationarity is a crucial assumption in time series analysis. A stationary time series exhibits constant statistical properties over time, such as a constant mean, variance, and auto covariance. It simplifies the modeling process and allows for meaningful interpretations.

If a time series is non-stationary, differencing can be applied to make it stationary. Differencing involves subtracting the previous observation from the current observation. It removes trends and seasonality, making the series stationary and suitable for modeling with ARIMA.

ARIMA model can help businesses optimize inventory management, production planning, and resource allocation. It provides insights into sales patterns, demand fluctuations, and potential factors influencing sales performance.

ARIMA models can be enhanced by incorporating external factors like marketing campaigns, economic indicators, or competitor data, through the SARIMAX framework. This allows for a more comprehensive analysis and accurate sales forecasts.

# 4.2 Neural Network

## 4.2.0 What are supervised learning algorithms.

When we train the algorithm by providing the labels explicitly, it is known as supervised learning. This type of algorithm uses the available dataset to train the model. The objective of Supervised Machine Learning Algorithms is to find the hypothesis as approx. as possible so that when there is new input data, the output y can be predicted. The application of supervised machine learning is to predict whether a mail is spam or not spam or face unlock in your smartphone. Supervised Machine Learning is divided into two parts based upon their output:

1. Regression
2. Classification

In Regression the output variable is numerical(continuous) i.e., we train the hypothesis(f(x)) in a way to get continuous output(y) for the input data(x). Since the output is informed of the real number, the regression technique is used in the prediction of quantities, size, values, etc.
For Example, we can use it to predict the price of the house given the dataset containing the features of the house like area, floor, etc.

In classification, the output variable is discrete. i.e., we train the hypothesis(f(x)) in a way to get discrete output(y) for the input data(x). The output can also be termed as a class. For example, by taking the above example of house price, we can use classification to predict whether the house price will be above or below instead of getting the exact value. So we have two classes, one if the price is above and the other if it is below. Classification is used in speech recognition, image classification, NLP, etc.

## 4.2.1 Multi-Layer perceptron

Multi-layer perceptron (MLP) is a supplement of feed forward neural network. It consists of three types of layers—the input layer, output layer and hidden layer, as shown in Fig. below. The input layer receives the input signal to be processed. The required task such as prediction and classification are performed by the output layer. An arbitrary number of hidden layers that are placed in between the input and output layer are the true computational engine of the MLP. Like a feed forward network in a MLP the data flows in the forward direction from input to output layer. The neurons in the MLP are trained with the back propagation learning algorithm. MLPs are designed to approximate any continuous function and can solve problems which are not linearly separable. The major use cases of MLP are pattern classification, recognition, prediction and approximation.

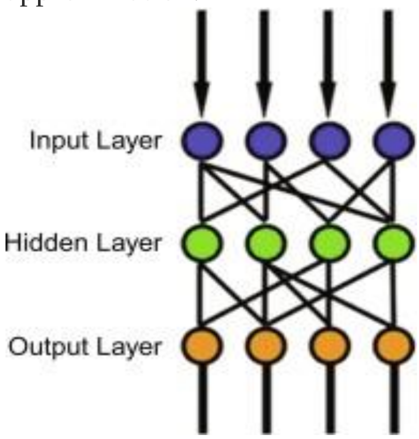


fig. 4.21 MLP

## 4.2.2 How does NNA learn.

Neural networks are generating a lot of excitement, as they are quickly proving to be a promising and practical form of machine intelligence. At Fast Forward Labs, we just finished a project researching and building systems that use neural networks for image analysis, as shown in our toy application Pictograph. Our companion deep learning report explains this technology in depth and explores applications and opportunities across industries. Each neuron in a network transforms data using a series of computations: a neuron multiplies an initial value by some weight, sums results with other values coming into the same neuron, adjusts the resulting number by the neuron's bias, and then normalizes the output with an activation function. The bias is a neuron-specific number that adjusts the neuron's value once all the connections are processed, and the activation function ensures values that are passed on lie within a tunable, expected range. This process is repeated until the final output layer can provide scores or predictions related to the classification task at hand, e.g., the likelihood that a dog is in an image.

### 4.2.3 Activation function.

The activation function decides whether a neuron should be activated or not by calculating the weighted sum and further adding bias to it. The purpose of the activation function is to introduce non-linearity into the output of a neuron. We know, the neural network has neurons that work in correspondence with weight, bias, and their respective activation function. In a neural network, we would update the weights and biases of the neurons based on the error at the output. This process is known as back-propagation. Activation functions make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases. In artificial neural networks, the activation function of a node defines the output of that node or neuron for a given input or set of inputs. This output is then used as input for the next node and so on until a desired solution to the original problem is found.

It maps the resulting values into the desired range such as between 0 to 1 or -1 to 1 etc. It depends upon the choice of the activation function. For example, the use of the logistic activation function would map all inputs in the real number domain into the range of 0 to 1.

### 4.2.4 Loss function.

Loss functions are used to determine the error (aka “the loss”) between the output of our algorithms and the given target value. In layman’s terms, the loss function expresses how far off the mark our computed output is. There are multiple ways to determine loss. Two of the most popular loss functions in machine learning are the 0-1 loss function and the quadratic loss function. The 0-1 loss function is an indicator function that returns 1 when the target and output are not equal and zero otherwise.
The quadratic loss is a commonly used symmetric loss function. The quadratic losses’ symmetry comes from its output being identical with relation to targets that differ by some value x in any direction (i.e., if the output overshoots by 1, that is the same as undershooting by 1). The quadratic loss is of the following form:

$$\text{Quadratic Loss: } (y, \hat{y}) = C\,(y- \hat{y})2$$

In the formula above, C is a constant and the value of C has made no difference to the decision. C can be ignored if set to 1 or, as is commonly done in machine learning, set to ½ to give the quadratic loss a nice differentiable form.

### 4.2.5 Learning rate

In supervised learning, to enable an algorithm's predictions to be as close to the actual values/labels as possible, we employ two things:
1) A cost function and
2) A technique to minimize the cost function. There are popular forms of cost functions used for different tasks that the algorithms are expected to perform.

Also, a popular technique used to minimize the cost function is the gradient descent method. We will understand these concepts to understand the role of 'learning rate' in machine learning. In machine learning, we deal with two types of parameters:
1) machine learnable parameters and
2) hyper-parameters.

The Machine learnable parameters are the one which the algorithms learn/estimate on their own during the training for a given dataset. In equation 3, $\beta 0$, $\beta 1$ and $\beta 2$ are the machine learnable parameters. The Hyper-parameters are the one which the machine learning engineers or data scientists will assign specific values to, to control the way the algorithms learn and to tune the performance of the model. Learning rate, generally represented by the symbol '$\alpha$', shown in equation 4, is a hyper-parameter used to control the rate at which an algorithm updates the parameter estimates or learns the values of the parameters.

## 4.3 Long Short-Term Memory (LSTM)

### 4.3.0 The architecture of a Recurrent Neural Network (RNN)

The architecture of a Recurrent Neural Network (RNN) is designed to process sequential data by incorporating a feedback loop, allowing information to persist and flow from one step to the next. The core element of an RNN is the recurrent connection, which enables the network to maintain memory of previous inputs and propagate information through time.

The basic architecture of an RNN consists of the following components:

Input: At each time step, the RNN receives an input vector representing the current information or observation in the sequence.

Hidden State: The hidden state is a vector that captures the network's memory or representation of the sequential information processed so far. It serves as a summary of the past inputs and influences the computation at the current time step. The hidden state is updated and passed from one-time step to the next, enabling the network to retain information about the sequence.

Recurrent Connection: The recurrent connection connects the hidden state of the current time step with the hidden state of the previous time step, forming a feedback loop. This connection allows the RNN to leverage the information from previous steps and use it to influence the computation at the current step. The recurrent connection is a key distinguishing feature of an RNN that enables it to handle sequential data effectively.

Output: The RNN produces an output vector at each time step, which can represent predictions, classifications, or any other desired output based on the task at hand. The output may depend on the current input and the hidden state.

The basic architecture described above is often referred to as a "vanilla" RNN or a simple RNN. However, there are variations and enhancements to this basic architecture, such as Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) networks, which introduce additional gates and mechanisms to improve the handling of long-term dependencies and mitigate issues like vanishing gradients. These variations in RNN architecture provide improved capabilities to capture and retain information over longer sequences, making them more effective in tasks involving sequential data.

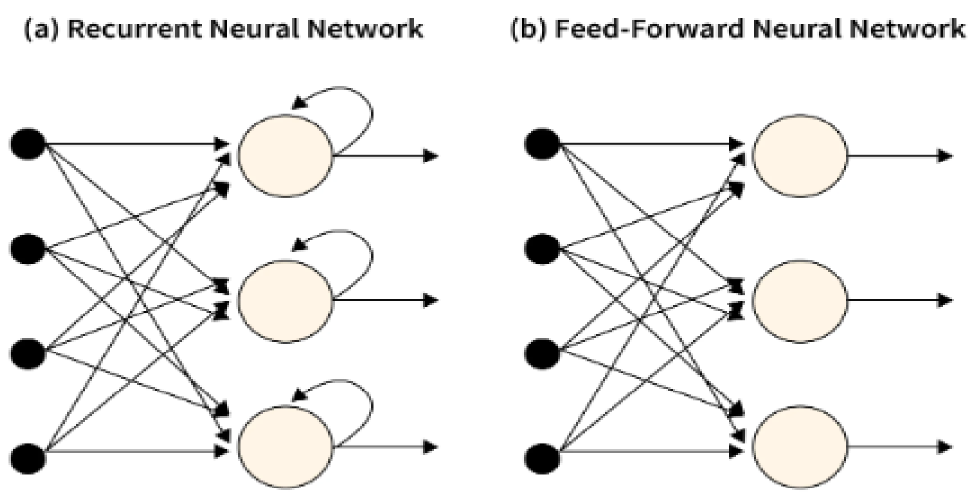


fig 4.30. FFNN vs RNN

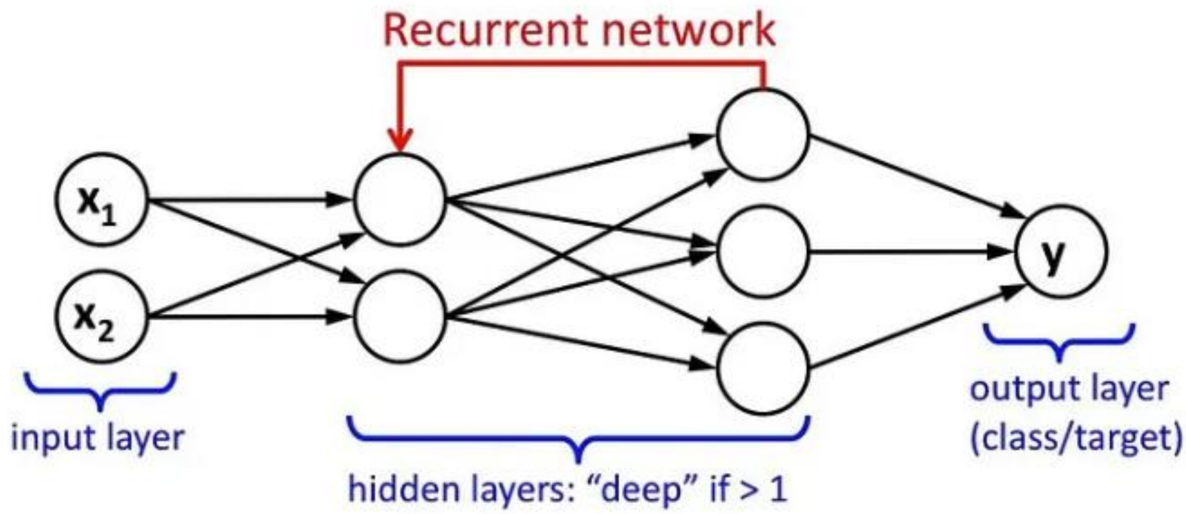


fig 4.30. RNN

## 4.3.1 LSTM

LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) architecture that overcomes the limitations of traditional RNNs in capturing and retaining long-term dependencies in sequential data. It is specifically designed to address the vanishing gradient problem, where gradients diminish exponentially during backpropagation through time in RNNs.

The main differences between LSTM and RNN are as follows:

Memory Cells: LSTMs have dedicated memory cells that allow them to store and access information over multiple time steps. This enables LSTMs to retain long-term memory and capture dependencies over extended sequences. In contrast, traditional RNNs do not have explicit memory cells and struggle to maintain information over long distances in the sequence.

Gating Mechanisms: LSTMs incorporate gating mechanisms, such as the forget gate, input gate, and output gate, to regulate the flow of information within the network. These gates control the extent to which information is stored, discarded, and exposed to the next hidden state. Gating mechanisms provide LSTMs with the ability to selectively update and utilize information, enhancing their capacity to handle long-term dependencies. In standard RNNs, there are no such gating mechanisms.

Handling Gradient Flow: LSTMs are designed to mitigate the vanishing gradient problem encountered in traditional RNNs. By allowing information to flow unchanged through the cell state and introducing gating mechanisms, LSTMs can effectively propagate gradients over long sequences without their magnitudes diminishing exponentially. This ensures that information from earlier time steps can influence the training process and contribute to capturing long-term dependencies.

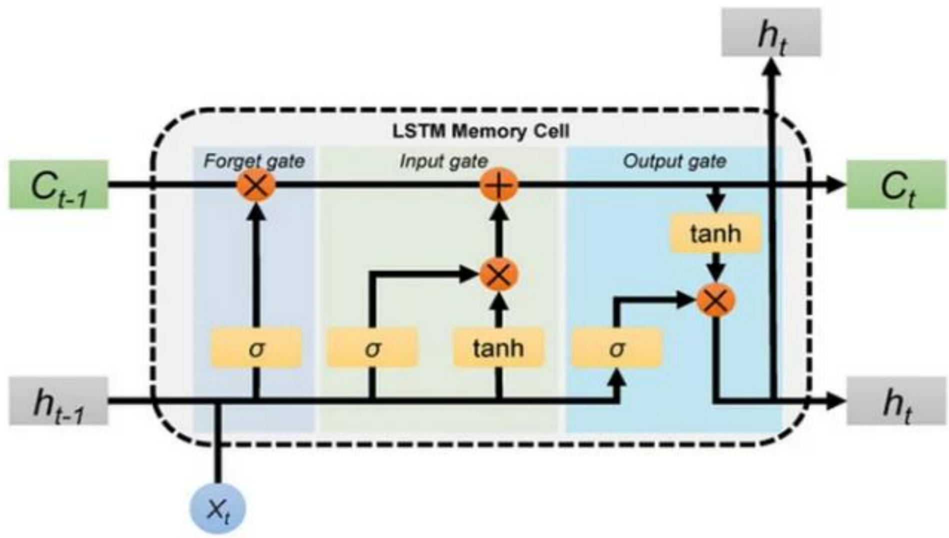


fig.4.31 LSTM

## 4.3.2 Types of LSTM

**a)** Stacked LSTM with 1 hidden dense layer. Stacked LSTM refers to the stacking of multiple LSTM layers on top of each other. In this configuration, the output sequence of one LSTM layer becomes the input sequence for the next LSTM layer. By stacking multiple LSTM layers, the network can learn more complex representations and capture higher-level dependencies in the sequential data. Adding a hidden dense layer after the stacked LSTM layers introduces a fully connected layer that performs non-linear transformations on the LSTM outputs, potentially enhancing the model's ability to learn intricate patterns and make predictions.

**b)** Bidirectional LSTM is a variation of LSTM that processes the input sequence in both forward and backward directions. By running two separate LSTM layers simultaneously, one in the forward direction and the other in the backward direction, the model can capture dependencies from past and future context. This allows the network to have a more comprehensive understanding of the sequential data and extract relevant features from both directions. Bidirectional LSTM is commonly used in tasks where future information is important, such as speech recognition, natural language processing, and sentiment analysis.

**c)**ConvLSTM combines the concepts of Convolutional Neural Networks (CNNs) and LSTM. It introduces convolutions within the LSTM structure, allowing the model to leverage the spatial structure of the input sequence. ConvLSTM is particularly useful for processing spatio-temporal data, such as videos or sequences with 2D spatial dimensions. The convolutional operations in ConvLSTM enable the network to capture local patterns and extract relevant spatial features from the input sequence, while the LSTM components maintain memory and capture temporal dependencies. ConvLSTM has shown promising results in video analysis, action recognition, and weather prediction tasks.

These three types of LSTM architectures, stacked LSTM with a hidden dense layer, bidirectional LSTM, and ConvLSTM, offer additional capabilities and architectural enhancements to the basic LSTM model. They address specific requirements and challenges in different domains and applications, allowing for more effective and accurate modeling of sequential data.

# CHAPTER 5

## 5.0 introduction

The implementation stage in a sales forecasting project encompasses the operationalization and integration of the developed forecasting model into the existing sales forecasting process. This phase focuses on translating the theoretical model into a practical tool that generates precise and dependable sales forecasts.

Successful implementation of the sales forecasting model transforms it from a theoretical concept into a practical tool. By integrating the model into existing systems, ensuring data availability and quality, automating processes, monitoring performance, and facilitating effective communication and collaboration among stakeholders, businesses can make informed decisions, optimize resource allocation, and enhance overall sales forecasting accuracy.

## 5.1 Steps of implementation

the first step is loading all the required libraries and upload the dataset

**1-Prepare the data before building the model**

We have appended all the 3 categories of the dataset into one sheet then we group the sales of each category by the weekly date. We have split the data into X set "input and Y set "output of the neural network.

**2-Normalization**

We have applied normalization for X set and Y set to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information

The most popular normalization technique is Rescaling (min-max normalization): scales your values information. in the range (0, 1). This transform features by scaling each feature to a given range.

**3-Data Splitting**

We split the X set into (X_train, X_test) and split Y set into (Y train,Y_test). This procedure has one main configuration parameter, which is the size of the train and test sets. This is most commonly expressed as a percentage between 0 and 1 for either the train or test datasets. Here the test set with the size of 0.30 (30 percent) means that the remainder percentage 0.70 (70 percent) is assigned to the training set.

**4-Model Definition**

<u>ARIMA Model</u>

The ARIMA model combines three components - auto regression (AR), differencing (I), and moving average (MA) - to capture the underlying patterns and dynamics in time series data.so we assume this component in this order (0,1,1)

MLP Model

We define the MLP model and its parameters:

Input dimension: The total sales for each week "40 neurons " Layers: we have one hidden layer consists of 32 neurons and output layer consists of 1 neuron, output represents weekly sales for a category Loss function: mean square error.

LSTM Model

We define the LSTM model and its parameters:

Input dimension: The total sales for each week "50 neurons " Layers: we have two hidden layer consists of 50 and 32 neurons and output layer consists of 1 neuron, output represents weekly sales for a category Loss function: mean square error.

**5-Training Model**

ARIMA Model

The training process involves selecting the appropriate values for the orders of auto regression (p), differencing (d), and moving average (q) components based on analysis of the data, such as autocorrelation and partial autocorrelation plots. Once the orders are determined, the model is fitted to the training set using algorithms like maximum likelihood estimation (MLE) or least squares estimation (LSE). The parameter estimates are obtained by minimizing the difference between the observed data and the predicted values generated by the ARIMA model.

MLP Model

Neural networks require a sufficient amount of training data to effectively learn patterns and make accurate predictions. The training process involves forward propagation, where the input data is passed through the network's layers, and an output is generated using randomly initialized weights and biases. This output is then compared to the actual output, and an error is calculated.

LSTM Model

After the forward propagation, the generated output is compared to the desired output, and an error is computed. This error is used to calculate a loss function that quantifies the discrepancy between the predicted and actual outputs. The goal of model training is to minimize this loss function.

To update the LSTM model's parameters, the backpropagation through time (BPTT) algorithm is utilized. Backpropagation involves propagating the error gradients backward through the LSTM layers to adjust the weights and biases. The gradients are computed using the chain rule and updated using an optimization algorithm, such as gradient descent or its variants.

During training, the LSTM model undergoes multiple iterations or epochs, where the entire dataset is processed. In each epoch, the model fine-tunes its weights and biases based on the gradients computed during backpropagation. The training continues until the model reaches a satisfactory level of performance or when a stopping criterion, such as a maximum number of epochs or early stopping, is met.

**6-Test Model**

we check the model's accuracy by testing it on new data that has never been used for training before. This helps to understand how each model performs in the real world. We used testing data "X test" for testing. We calculate the mean square error of testing and are it with the mean square error of training then we plot the cost function of training and Our goal to get testing error close as possible to training error.

## 5.2 Results

## 5.2.0 Model Validation

Model validation is a crucial step in assessing the performance and reliability of a machine learning or statistical model. It involves evaluating the model's predictive ability and generalization to unseen data.

1.Training and Testing Data: Model validation starts by dividing the available dataset into training and testing sets. The training set is used to train the model, while the testing set is used to evaluate its performance on unseen data.

2.Evaluation Metrics: Various evaluation metrics are used to quantify the performance of the model. Common metrics include accuracy, precision, recall, F1 score, mean squared error, and others. The choice of metric depends on the specific problem and the nature of the data.

3.Model Comparison: Model validation enables the comparison of different models or algorithms to select the best-performing one. Multiple models can be trained and evaluated using the same validation protocol and metrics. This allows for an objective comparison of their performance and the selection of the most suitable model for the task.

Overall, model validation is a critical step to ensure that the developed model performs well on unseen data, avoids overfitting or under fitting, and meets the desired performance criteria. It helps build confidence in the model's reliability and aids in making informed decisions based on its predictions.

### 5.2.1 Evaluation metrics

Evaluation metrics are quantitative measures used to assess the performance and accuracy of machine learning models or forecasting algorithms. These metrics provide objective criteria for evaluating the model's predictions or classifications. Here is a summary of common evaluation metrics:

1.Accuracy: Measures the proportion of correctly classified instances or the percentage of correct predictions. It is commonly used for balanced classification tasks.

2.Mean Squared Error (MSE): Commonly used in regression tasks, it measures the average squared difference between predicted and actual values.

3.Mean Absolute Error (MAE): Calculates the average absolute difference between predicted and actual values. It provides a more interpretable measure of error than MSE.

4.R-squared (Coefficient of Determination): Measures the proportion of the variance in the dependent variable explained by the independent variables in a regression model.

These evaluation metrics help assess the model's performance and guide the selection and optimization of machine learning models. The choice of evaluation metric depends on the specific problem, data characteristics, and project goals. It is important to select metrics that align with the objectives and requirements of the analysis to accurately assess the model's effectiveness.

## 5.2.2 Results Discussion

| Model Name | Mean Squared Error (MSE) |
|---|---|
| ARIMA | 0.37038 |
| MLP | 0.17937 |
| LSTM | 0.21238 |

Tab3 Result Discussion

Based on the MSE results, it can be concluded that the MLP model had the lowest average squared difference between the predicted and actual values, indicating the highest level of accuracy among the three models. However, the specific implications and significance of the MSE values would depend on the particular dataset, its characteristics, and the specific goals of the modeling task. It is always important to consider other evaluation metrics and domain knowledge to comprehensively assess the performance of the models.

# CHAPTER 6

## Conclusion

In this retail sales forecasting project, we aimed to develop accurate and reliable models to forecast sales in the retail industry. By analyzing historical sales data and applying various forecasting techniques, we have gained valuable insights and achieved significant results.

Firstly, we conducted exploratory data analysis to understand the characteristics and patterns within the retail sales data. We identified trends, seasonality, and other factors that impact sales fluctuations. This analysis helped us tailor our modeling approaches to address the specific features of the dataset.

Next, we implemented and compared different forecasting models, including traditional statistical models like ARIMA, machine learning models like MLP, and deep learning models like LSTM. Each model brought its strengths and limitations to the table.

Based on our evaluation metrics, such as Mean Squared Error (MSE), we found that the MLP model outperformed the other models, achieving the lowest MSE value. This indicates that the MLP model had a higher accuracy in predicting retail sales compared to the ARIMA and LSTM models. The MLP model's ability to capture complex nonlinear relationships in the data proved advantageous in forecasting sales patterns.

However, it is important to note that the choice of the best model depends on the specific requirements of the retail industry and the characteristics of the dataset. Different models may excel in different scenarios, and it is crucial to consider other evaluation metrics, domain knowledge, and practical constraints when making the final model selection.

Our project highlights the importance of data analysis, model selection, and continuous evaluation to achieve accurate retail sales forecasting. Accurate forecasts enable retailers to optimize inventory management, plan marketing strategies, and make informed business decisions to meet customer demand effectively.

Furthermore, this project emphasizes the need for continuous monitoring and updating of forecasting models. As retail sales data evolves, models should be retrained periodically to adapt to changing patterns and dynamics in the market. Regular evaluation and refinement of the forecasting process are necessary to maintain accurate and up-to-date predictions.

In conclusion, this retail sales forecasting project demonstrates the value and potential of implementing advanced modeling techniques in the retail industry. By leveraging the power of data analysis and predictive modeling, retailers can gain a competitive edge, improve decision-making, and enhance operational efficiency. However, ongoing evaluation, refinement, and adaptation are essential to ensure the models remain robust and reliable in an ever-evolving retail landscape.

# Appendix – I

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from pmdarima.arima import auto_arima
from statsmodels.tsa.statespace.sarimax import SARIMAX
import math
from keras.models import Sequential
from keras.layers import Dense
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error , r2_score
from keras.layers import LSTM, Flatten

data =pd.read_csv('Cairo.csv')
data.shape

data.head()

data.info()

data['PRODUCT_CODE'] = data['PRODUCT_CODE'].astype(str)

data.describe()

"""Data Exploration"""

subcategory_sales = data.groupby('Category')['Total Price'].sum()
subcategory_sales.plot(kind='pie')
plt.xlabel('Category')
plt.ylabel('Total Sales')
plt.show()

subcategory_sales = data.groupby('subcategory')['Total Price'].sum()
subcategory_sales.plot(kind='bar')
plt.xlabel('Subcategory')
plt.ylabel('Total Sales')
plt.show()

correlation_matrix = data[['Quantity', 'Price_Per_Piece', 'Total Price']].corr()
plt.imshow(correlation_matrix, cmap='coolwarm', interpolation='nearest')
plt.colorbar()
plt.xticks(range(correlation_matrix.shape[1]), correlation_matrix.columns, rotation=45)
plt.yticks(range(correlation_matrix.shape[1]), correlation_matrix.columns)
plt.show()
```

```
"""Data Manipulation"""

#Filter by category
category_name = 'Soap'
filtered_df = data[data['Category'] == category_name]

columns_to_drop = ['PRODUCT_CODE', 'Category','Price_Per_Piece', 'Quantity','subcategory']
filtered_df = data.drop(columns=columns_to_drop)

filtered_df.info()

filtered_df

#Time series analysis
filtered_df['Date'] = pd.to_datetime(filtered_df['Date'])
filtered_df.set_index('Date', inplace=True)
weekly_sales = filtered_df['Total Price'].resample('W').sum()
plt.plot(weekly_sales)
plt.xlabel('Week')
plt.ylabel('Total Sales')
plt.show()

weekly_sales = pd.DataFrame(weekly_sales)
```

```
"""ARIMA Model"""

arima_model = auto_arima(weekly_sales['Total Price'], start_p = 1, d=1, start_q = 1,
                          max_p = 5, max_q = 5, max_d=5, m = 12,
                          start_P = 0, D=1, start_Q=0, max_P=5, max_D=5, max_Q=5,
                          seasonal = True,
                          trace = True,
                          error_action ='ignore',
                          suppress_warnings = True,
                          stepwise = True, n_fits=50)

print(arima_model.summary() )

size = int(len(weekly_sales) * 0.66)
X_train, X_test = weekly_sales[0:size], weekly_sales[size:len(weekly_sales)]

model = SARIMAX(X_train['Total Price'],
                order = (0, 1, 3),
                seasonal_order =(0, 1, 1, 12))

result = model.fit()
result.summary()


#Train prediction
start_index = 0
end_index = len(X_train)-1
train_prediction = result.predict(start_index, end_index)

#Prediction
start_index = len(X_train)
end_index = len(weekly_sales)-1
prediction = result.predict(start_index, end_index).rename('Predicted Total Price')
#Rename the column
```

```python
import math
from sklearn.metrics import mean_squared_error
# calculate root mean squared error and R^2
trainScore = math.sqrt(mean_squared_error(X_train, train_prediction))
print('Train Score: %.2f RMSE' % (trainScore))
testScore = math.sqrt(mean_squared_error(X_test, prediction))
print('Test Score: %.2f RMSE' % (testScore))

score = r2_score(X_test, prediction)
print("R2 score is: ", score)


forecast = result.predict(start = len(weekly_sales),
                          end = (len(weekly_sales)-1) + 3 * 12,
                          typ = 'levels').rename('Forecast')
# plot predictions and actual values
prediction.plot(legend = True)
X_test['Total Price'].plot(legend = True)
```

```python
"""Prepare the data for FFNN Model and LSTM Model

Normalization
"""

scaler = MinMaxScaler(feature_range=(0, 1))
weekly_sales = scaler.fit_transform(weekly_sales)

"""Data Splitting"""

train_size = int(len(weekly_sales) * 0.66)
test_size = len(weekly_sales) - train_size
train, test = weekly_sales[0:train_size,:], weekly_sales[train_size:len(weekly_sales),:]
def to_sequences(weekly_sales, seq_size=1):
    x = []
    y = []

    for i in range(len(weekly_sales)-seq_size-1):
        #print(i)
        window = weekly_sales[i:(i+seq_size), 0]
        x.append(window)
        y.append(weekly_sales[i+seq_size, 0])

    return np.array(x),np.array(y)
seq_size = 10 # Number of time steps to look back
trainX, trainY = to_sequences(train, seq_size)
testX, testY = to_sequences(test, seq_size)
print("Shape of training set: {}".format(trainX.shape))
print("Shape of test set: {}".format(testX.shape))
```

```
"""FFNN"""

# create and fit dense model
model = Sequential()
model.add(Dense(64, input_dim=seq_size, activation='relu')) #12
model.add(Dense(32, activation='relu'))  #8
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam', metrics = ['acc'])
print(model.summary())

#Training Model
model.fit(trainX, trainY, validation_data=(testX, testY),
          verbose=2, epochs=500)

# make predictions
trainPredict = model.predict(trainX)
testPredict = model.predict(testX)
trainPredict = scaler.inverse_transform(trainPredict)
trainY_inverse = scaler.inverse_transform([trainY])
testPredict = scaler.inverse_transform(testPredict)
testY_inverse = scaler.inverse_transform([testY])

#Evaluation Model
trainScore = math.sqrt(mean_squared_error(trainY_inverse[0], trainPredict[:,0]))
print('Train Score: %.2f RMSE' % (trainScore))
testScore = math.sqrt(mean_squared_error(testY_inverse[0], testPredict[:,0]))
print('Test Score: %.2f RMSE' % (testScore))

# invert predictions back to prescaled values
trainPredictPlot = np.empty_like(weekly_sales)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[seq_size:len(trainPredict)+seq_size, :] = trainPredict
testPredictPlot = np.empty_like(weekly_sales)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(trainPredict)+(seq_size*2)+1:len(weekly_sales)-1, :] = testPredict

# plot baseline and predictions
plt.plot(scaler.inverse_transform(weekly_sales))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
```

```
"""LSTM"""

#Stacked LSTM with 1 hidden dense layer
#reshape input to be [samples, time steps, features]
trainX = np.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
testX = np.reshape(testX, (testX.shape[0], 1, testX.shape[1]))

model = Sequential()
model.add(LSTM(50, activation='relu', return_sequences=True, input_shape=(None, seq_size)))
model.add(LSTM(50, activation='relu'))
model.add(Dense(32))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')

model.summary()

#Training Model
model.fit(trainX, trainY, validation_data=(testX, testY),
          verbose=2, epochs=500)


# make predictions

trainPredict = model.predict(trainX)
testPredict = model.predict(testX)

# invert predictions back to prescaled values
trainPredict = scaler.inverse_transform(trainPredict)
trainY = scaler.inverse_transform([trainY])
testPredict = scaler.inverse_transform(testPredict)
testY = scaler.inverse_transform([testY])
```

```
# Evaluation Model
trainScore = math.sqrt(mean_squared_error(trainY[0], trainPredict[:,0]))
print('Train Score: %.2f RMSE' % (trainScore))

testScore = math.sqrt(mean_squared_error(testY[0], testPredict[:,0]))
print('Test Score: %.2f RMSE' % (testScore))

# shift train predictions for plotting
#we must shift the predictions so that they align on the x-axis with the original dataset.
trainPredictPlot = np.empty_like(weekly_sales)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[seq_size:len(trainPredict)+seq_size, :] = trainPredict

# shift test predictions for plotting
testPredictPlot = np.empty_like(weekly_sales)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(trainPredict)+(seq_size*2)+1:len(weekly_sales)-1, :] = testPredict

# plot baseline and predictions
plt.plot(scaler.inverse_transform(weekly_sales))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
```

# Appendix – II

| Title | Date | Methodology | Conclusion |
|---|---|---|---|
| Effective Demand Forecasting Model Using Business Intelligence Empowered with Machine Learning | 2020 | We test the efficiency by comparing the predicted data with actual data and determine the percentage error. | Depending upon accuracy new study can be related to stock optimization. So, the stock/product optimization point can be a new starting point. |
| Machine Learning Approach for Forecasting the Sales of Truck Components | 2019 | A literature review is used to find suitable machine learning algorithms and then based on the results obtained, an experiment is performed to evaluate the performances of machine learning algorithms. | Based on the results obtained, four machine learning models namely Ridge Regression, Support Vector Regression and Gradient Boosting Regression have been chosen for forecasting the sales. |
| Multi-Layer Neural Networks for Sales Forecasting | 2018 | We used a feedforward artificial Neural Network trained on past data with the backpropagation algorithm. | - In the paper, we predict monthly sales volume of a textile warehouse.<br>- The main drawback of the presented method is the lack of the interpretability of the trained Neural Network as it acts as a black box. |

| | | | |
|---|---|---|---|
| Car Sales Forecasting Using Artificial Neural Networks and Analytical Hierarchy Process. | 2016 | Analytical Hierarchy Process (AHP) | Based on the positive performance of the neural network and fitting curves for subjects with limited and poor data. |
| Demand Forecasting Using Neural Network for Supply Chain Management | 2013 | The research uses Neural Network technique to investigate the influence of demand forecasting to predictions of next year consumptions. | The result indicates a Train LM method performs more effectively than the other tanning method and the reliable forecast for our case. The proposed methodology can be considered as a successful decision support tool in forecasting. The ability to increase forecasting accuracy will result. Future research can possibility of using Artificial Neural Network to make a similar approach and better the accuracy. |
| A Neural Network Model for Forecasting Production Time Series in Brazilian Industries | 2010 | Measure of forecast error in time series analysis. | For future works, we suggest that new models and algorithms are tested in order to find a better efficiency in the result presented. |

# REFRENCES


[1] https://mjee.isfahan.iau.ir/article_696444_18b88cd6f864ef3aa72a2af41f417365.pdf

[2]https://www.sciencedirect.com/science/article/pii/S2667096822000027?ref=pdf_download&fr=RR-2&rr=7dfdb9884f2e0777

[3]file:///C:/Users/DELL/Downloads/s10462-018-9637-z.pdf

[5]https://iopscience.iop.org/article/10.1088/1742-6596/1712/1/012042/pdf

[6]https://www.sciencedirect.com/science/article/pii/S240589631931153X

[7]https://intellipaat.com/blog/what-is-lstm/?US

[8]https://towardsdatascience.com/implementation-of-rnn-lstm-and-gru-a4250bf6c090

[9]https://www.scaler.com/topics/deep-learning/rnn/

[10]https://www.turing.com/kb/comprehensive-guide-to-time-series-analysis-in-Python

[11] https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/

[12] https://medium.com/@b.terryjack/introduction-to-deep-learning-feed-forward-neural-networks-ffnns-a-k-a-c688d83a309d

[13] https://machinelearningmastery.com/types-of-learning-in-machine-learning/

[14] https://www.coursera.org/articles/machine-learning-models

[15] https://link.springer.com/chapter/10.1007/978-981-15-0199-9_11

[16] https://www.sciencedirect.com/science/article/abs/pii/S0360835220304125