

*A Mini Project Report on*  
**ARIMA MODEL FOR PREDICTION OF VEHICLE  
SALES**

*Submitted to*

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY,  
HYDERABAD**

*In Partial fulfillment of the requirement for the award of degree of the*

**BACHELOR OF TECHNOLOGY  
IN  
COMPUTER SCIENCE AND ENGINEERING**

*By*

<b>K. VISHNUVARDHAN REDDY</b>	<b>19RA1A0553</b>
<b>N. SAI</b>	<b>19RA1A0532</b>
<b>M. UDAY</b>	<b>19RA1A0547</b>

*Under the guidance of*

**G. SUJATHA**  
(Assistant Professor)

**Department of Computer Science and Engineering**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY  
(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-506345)**

**2019-2023**

# KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-506345)



## CERTIFICATE

This is to certify that the Mini Project Report entitled “**ARIMA Model for Prediction of Vehicle Sales**” is submitted by Mr. K. Vishnuvardhan Reddy, Mr. N. Sai, Mr. M. Uday, bonafided student of **Kommuri Pratap Reddy Institute of Technology** in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology in **Computer Science and Engineering** of the **Jawaharlal Nehru Technological University, Hyderabad** during the year 2019-23.

**Internal Examiner**  
**Mrs. G. Sujatha**

**HOD**  
**Mrs. G. Sujatha**

**External Examiner**

## **DECLARATION**

We hereby declare that this project work entitled “**ARIMA Model for Prediction of Vehicle Sales**” in partial fulfillment of requirements for the award of degree of **Computer Science and Engineering** is a bonafided work carried out by us during the academic year 2019- 23.

We further declare that this project is a result of our effort and has not been submitted for the award of any degree by us to any institute.

*By*

K. Vishnuvardhan Reddy (19RA1A0553)

N. Sai (19RA1A0532)

M. Uday (19RA1A0547)

## ACKNOWLEDGEMENT

It gives us immense pleasure to acknowledge with gratitude, the help and support extended throughout the project report from the following:

We will be very much grateful to almighty our **Parents** who have made us capable of carrying out our job.

We express our profound gratitude to **Dr. D. Eshwar, Principal of Kommuri Pratap Reddy Institute of Technology**, who has encouraged in completing our project report successfully.

We are grateful to **Mrs. G. Sujatha** who is our **Head of the Department, CSE** for her amiable ingenious and adept suggestions and pioneering guidance during the project report.

We express our gratitude and thanks to the coordinator **Dr. C. Veena** of our department for her contribution for making it success with in the given time duration.

We express our deep sense of gratitude and thanks to **Internal Guide Mrs. G. Sujatha Assistant Professor and Professor** for her guidance during the project report.

We are also very thankful to our Management, Staff Members and all Our Friends for their valuable suggestions and timely guidance without which we would not have been completed it.

*By*

**K. Vishnuvardhan Reddy (19RA1A0553)**

**N. Sai (19RA1A0532)**

**M. Uday (19RA1A0547)**

## **Vision of the Institute**

To emerge as a premier institute for high quality professional graduates who can contribute to economic and social developments of the Nation.

## **Mission of Institute**

<b>Mission</b>	<b>Statement</b>
<b>DM1</b>	Laying the path for rich skills in Computer Science through The basic knowledge of mathematics and fundamentals of engineering
<b>DM2</b>	Provide latest tools and technology to the students as a part of learning Infrastructure
<b>DM3</b>	Training the students towards employability and entrepreneurship to meet the societal needs.
<b>DM4</b>	Grooming the students with professional and social ethics.

***Vision of the Department***

To Provide Quality Education in Computer Science for the innovative professionals to work for the development of the nation.

**Mission of the Department**

<b>Mission</b>	<b>Statement</b>
<b>DM1</b>	Laying the path for rich skills in Computer Science through The basic knowledge of mathematics and fundamentals of engineering
<b>DM2</b>	Provide latest tools and technology to the students as a part of learning Infrastructure
<b>DM3</b>	Training the students towards employability and entrepreneurship to meet the societal needs.
<b>DM4</b>	Grooming the students with professional and social ethics.

### Program Educational Objectives (PEOs)

PEO's	Statement
<b>PEO1</b>	The graduates of Computer Science and Engineering will have successful career in technology.
<b>PEO2</b>	The graduates of the program will have solid technical and professional foundation to continue higher studies.
<b>PEO3</b>	The graduate of the program will have skills to develop products, offer services and innovation
<b>PEO4</b>	The graduates of the program will have fundamental awareness of industry process, tools and technologies.

### Program Outcomes

<b>PO1</b>	<b>Engineering Knowledge:</b> Apply the knowledge of mathematics, science, Engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
<b>PO2</b>	<b>Problem Analysis:</b> Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
<b>PO3</b>	<b>Design/development of Solutions:</b> Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
<b>PO4</b>	<b>Conduct investigations of complex problems:</b> Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
<b>PO5</b>	<b>Modern tool usage:</b> Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
<b>PO6</b>	<b>The engineer and society:</b> Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
<b>PO7</b>	<b>Environment and sustainability:</b> Understand the impact of the professional engineering solutions in societal and environmental context, and demonstrate the knowledge of, and need for sustainable development.



<b>PO8</b>	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
<b>PO9</b>	Individual and team network: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
<b>PO10</b>	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, being able to comprehend and write effective reports and design documentation, make Effective presentations, and give and receive clear instructions.
<b>PO11</b>	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in a multidisciplinary environment.
<b>PO12</b>	Life-Long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## PROGRAM SPECIFIC OUTCOMES

<b>PSO1</b>	Foundation of mathematical concepts: To use mathematical methodologies to crack problems using suitable mathematical analysis, data structure and suitable algorithms.
<b>PSO2</b>	Foundation of Computer Science: The ability to interpret the fundamental concepts and methodology of computer systems. Students can understand the functionality of hardware and software aspects of computer systems.
<b>PSO3</b>	Foundation of Software development: The ability to grasp the software development lifecycle and methodologies of software systems. Possess competent skills and knowledge of the software design process.

## **ABSTRACT**

This project presents enhanced sales forecast methodology and model for the automobile market which delivers highly accurate predictions while maintaining the ability to explain the underlying model at the same time. The representation of the economic training data is discussed, as well as its effects on the newly registered automobiles to be predicted. The methodology mainly consists of time series analysis and classical data mining algorithms, whereas the data is composed of absolute and/or relative market-specific exogenous parameters on a yearly, quarterly, or monthly base. It can be concluded that the monthly forecasts were especially improved by this enhanced methodology using absolute, normalized exogenous parameters. The main goal of this project is to consider main approaches and case studies of using machine learning for sales forecasting. The effect of machine-learning generalization has been considered. This effect can be used to make sales predictions when there is a small amount of historical data for specific sales time series in the case when a new product or store is launched. A stacking approach for building regression ensemble of single models has been studied. The results show that using stacking techniques, we can improve the performance of predictive models for sales time series forecasting.

## **TABLE OF CONTENTS**

ABSTRACT.....	I
LIST OF FIGURES.....	IV
LIST OF SCREENSHOTS.....	V
1. INTRODUCTION.....	1
2. LITERATURE SURVEY.....	3
3. SYSTEM ANALYSIS.....	6
3.1 SOFTWARE DEVELOPMENT LIFE CYCLE.....	6
3.1.1 REQUIREMENTS.....	6
3.1.2 DESIGN.....	7
3.1.3 IMPLEMENTATION.....	7
3.1.4 TESTING.....	7
3.2 EXISTING SYSTEM.....	7
3.2.1 SUPPORT VECTOR MACHINE (SVM).....	7
3.2.2 APPLICATIONS.....	8
3.2.3 DISADVANTAGES.....	9
3.3 PROPOSED SYSTEMS.....	9
3.3.1 ARIMA.....	9
3.3.2 DECOMPOSITION OF TIME SERIES.....	12
3.3.3 PRE-PROCESSING.....	13
3.3.4 TRAINING SET AND TEST SET.....	14
3.3.5 ADVANTAGES.....	14
4. SYSTEM DESIGN.....	15
4.1 UML DIAGRAM.....	15

4.1.1 USE CASE DIAGRAM.....	15
4.1.2 CLASS DIAGRAM.....	16
4.1.3 ACTIVITY DIAGRAM.....	16
4.1.4 SEQUENCE DIAGRAM.....	17
4.1.5 COMPONENT DIAGRAM.....	18
4.1.6 INTERACTION OVERVIEW DIAGRAM.....	18
5. SOFTWARE ENVIRONMENT.....	19
5.1 MACHINE LEARNING.....	19
5.2 PYTHON.....	26
6. SYSTEM REQUIREMENTS SPECIFICATION.....	35
6.1 SYSTEM REQUIERMENTS.....	35
6.1.1 SOFTWARE REQUIREMENT.....	35
6.1.2 HARDWARE REQUIREMENT.....	35
6.2 FUNCTIONAL REQUIREMENTS.....	35
6.2.1 OUTPUT DESIGN.....	35
6.2.2 INPUT DESIGN.....	36
7. SYSTEM IMPLEMENTATION.....	41
7.1 MODULES USED IN PROJECT.....	41
7.2 CODING.....	42
8. RESULTS.....	53
9. CONCLUSION.....	60
10. FUTURE SCOPE.....	61
11. REFERENCE.....	62

## LIST OF FIGURE

FIGURE NO.	FIGURE NAME	PAGE NUMBER
Figure 3.1	Software Development Life Cycle	6
Figure 3.2.1	Hyperplane	8
Figure 3.3	Block Diagram of Proposed System	9
Figure 3.3.1	Flow Diagram of ARIMA Model	10
Figure 3.3.4	Dataset	14
Figure 4.1.1	Use Case Diagram	15
Figure 4.1.2	Class Diagram	16
Figure 4.1.3	Activity Diagram	16
Figure 4.1.4	Sequence Diagram	17
Figure 4.1.5	Component Diagram	18
Figure 4.1.6	Interaction Overview Diagram	18

## **LIST OF SCREENSHOTS**

SCREENSHOT NO.	SCREENSHOT NAME	PAGE NUMBER
Screenshot 8.1	Original Data	53
Screenshot 8.2	Pre-Processing Data	54
Screenshot 8.3	Extracting Time Series Graph	54
Screenshot 8.4	Removing Wrinkles	55
Screenshot 8.5	Dickey-Fuller Test	55
Screenshot 8.6	Applying Log	56
Screenshot 8.7	Applying Difference log	56
Screenshot 8.8	ACF and PACF of Original Data	57
Screenshot 8.9	Predicting Sales on Sample Data	57
Screenshot 8.10	Best Fit ARIMA Model	58
Screenshot 8.11	Sale Prediction up to 2017	58
Screenshot 8.12	ACF and PACF of Predicted Sales	

# 1. INTRODUCTION

Sales prediction is a complex process and a challenging task for researchers. It includes expertise in multiple disciplines. The prediction of atmospheric parameters is essential for various applications. Accurate prediction of Sales parameters is a difficult task due to the dynamic nature of atmosphere. Generally, prediction is done on Time series data. A time series is a sequence of observed values of some entity that is measured at different points in time. With the advancement of collecting data, huge amounts of data have been collected making it impossible to be processed manually. This is where the time series analysis must be automated and take advantage of modern computing mechanisms. Various techniques like linear regression, auto regression, Multi-Layer Perceptron, Radial Basis Function networks are applied to predict atmospheric parameters like temperature, wind speed, rainfall, meteorological pollution etc. It was found that the nonlinear operator equations governing the atmospheric system are the ones who can better understand the dynamics of atmosphere. In the recent past many forecast methods have been developed using Artificial Neural Networks (ANNs). Neural network techniques have the potential to handle complex, nonlinear problems in a better way when compared to traditional techniques. However, systems developed using neural network model suffer from certain drawbacks like local minima, model over fitting etc. In this project we are going use ARIMA model for the predicting the sales.

Intelligent vehicles are a new generation of vehicles that are equipped with advanced sensors and other devices, use new technologies such as artificial intelligence, have self-driving functions, and gradually become intelligent mobile spaces and application terminals. In recent years, a new generation of industrial technology with intelligence and the Internet as the core has been on the rise, promoting traditional industries to accelerate the “intelligent” transformation and upgrade . As one of the most widespread and important industries, the automotive industry is also accelerating its progress of intelligence, and “Intelligent vehicles” are receiving more attention, especially in the areas of shared mobility, energy consumption, and vehicle safety . Most intelligent vehicles are electric vehicles, and the number of electric vehicles will continue to grow in the next 40 years with an “S” trend. The Chinese government has also made intelligent vehicles a priority to the development of the automotive industry, and the development strategy released by the country has drawn strong attention to the industry. Intelligent vehicles are increasingly becoming the focus of the automotive industry, but research related to intelligent vehicles is still relatively lacking.

Predicting sales is a key step in making production decisions on companies and public policy for governments. Companies use product sales predictions as a basis for estimating sales revenues. They can also use product sales predictions to develop plans for marketing, sales management, production, purchasing, and logistics to improve economic efficiency and reduce losses in production planning. Intense competition, large investments, and the need for rapid model updates characterize the automotive industry, which makes predicting crucial for sale and production processes. Most intelligent vehicle companies are emerging vehicle makers and have a low grasp of the market demand for their products, so it is more important to accurately predict the sales scale of intelligent vehicles for companies to set a reasonable production scale. In addition, as a strategic emerging industry supported by the Chinese government, the sales growth of intelligent vehicles is the result of the joint action between the market and the government. Therefore, the study of the intelligent vehicle market sales prediction is also important to the formulation of intelligent vehicle industry support policies and the arrangement of related supporting facilities.

Big data have become one of the important tools in the field of predicting and it has been very widely used in the traditional automotive field. Unlike traditional vehicles, intelligent vehicle brands come with Internet properties and are often also referred to as Internet vehicles, which is why they are widely followed and discussed on the Internet; with more than 6.9 million discussions about Tesla (Fremont, CA, USA), the intelligent vehicle brand, on the social platform Twitter from 2018 to 2020. In China, NIO, the emerging domestic intelligent vehicle brand, has 910,000 followers on the social platform Weibo, with more than 200,000 discussions about NIO on Weibo alone from 2018 to 2020. In this case, the brand's online public opinion and attention often affect the user's willingness to purchase and further affect the sales of the vehicle brand. In 2019, the domestic intelligent vehicle brand XPeng had a big drop in sales due to a collective online rights opinion storm, with sales down 43.9%. Previous studies have tended to focus only on traditional vehicle brands and less on the impact of online public opinion and online search index on intelligent vehicle sales. With the growing development of intelligent vehicles, it is urgent to establish a sales prediction model that applies to intelligent vehicles and considers the influence of online public opinion and attention.



## 2. LITERATURE SURVEY

Souza et al. aimed to estimate the future generation of ELVs to assist decision making and mitigate the global impact of this type of waste on the environment. For this, a hybrid forecasting model was used, based on Autoregressive Integrated Moving Average (ARIMA) methodology and on Artificial Neural Networks (ANN), with a set of temporal data extracted from Brazilian sectoral platforms. Considering the scarcity of information that supports decision-making in waste management in Brazil, this study may also contribute to the proposition of alternatives that favor the proper management of automotive waste, providing a reference for the formulation and implementation of policies related to ELVs in the country.

Mehendale et al. aimed at applying Artificial Intelligence (AI) techniques to identify and predict complex sales patterns and compare the results with traditional forecasting models. Sales data related to the sample firm was used in the study for forecast modelling using both, advanced forecasting method (ARIMA) and Artificial Neural Networks (ANNs). The latter considered inputs of influential/causal factors and revisits every time to identify new trends related to those factors, thus providing a more robust forecast. This is compared with results from the ARIMA model. Though the purpose of the research is achieved, a few limitations exist because of the limited availability of data. Besides, the factors that affect the prediction are generalized and based on previous research works. A more customized approach towards the firm under study would greatly improve the accuracy.

Nigam et al. presented Box-Jenkins method used to forecast the future demand in a two-wheeler industry. An automated technique in machine learning with the help of python language has been developed and used to analyze time series data and ultimately fit the model for future demand projection. The time series data is collected for the Royal Enfield bikes' monthly sale available at the official website of Eicher motors ltd. The resulting pattern found in time series data is used to forecast the future behavior, knowledge of which will help to maintain the appropriate inventory and to reduce the risk in terms of changing customers preferences, resource availability etc. Also, the effect of covid-19 pandemic has been captured to visualize its impact.

Swami et al. analyzed the trends of production, sales, and exports in the present paper with reference to the Indian automobile sector. The sales of vehicles in countries like Europe, America, Africa, NAFTA countries, and India have been considered to forecast the future sales.

Data for sales between the periods of 2005 to 2018 have been collected and analyzed using the ARIMA forecasting technique.

Shakti et al. used the ARIMA model for predicting the number of sales for a Time series data. The dataset tractor sales data for a period of ten years (2003-2014) obtained from the Mahindra Tractors Company are used from which used to classify the performance by drawing various scattered plots and graphs. The result of the ARIMA results showed that which predicted better for the sales prediction of the next following 5 years.

Irhami et al. obtained the best model for forecast the number of cars and the number of motorcycles in the next 11 years. For the purpose, ARIMA method was used. Using the historical data of the number of cars and the number of motorcycles from 2001 to 2019, the best model for forecasting the number of cars and the number of motorcycles is ARIMA (1,1,0) and ARIMA (2,1,2), respectively. The models have MAPE of 7.01% and 7.24% for cars and motorcycles, respectively.

Permatasari et al. predicted the printed newspaper demand as accurately as possible to minimize the number of returns, to keep off the missed sales and to restrain the oversupply. The autoregressive integrated moving average (ARIMA) models were adopted to predict the right number of newspapers for a real case study of a newspaper company in Surakarta. The model parameters were found using maximum likelihood method. Then, the software Eviews 9 were utilized to forecasting any variables in the newspaper industry. This project finally presented the appropriate of modeling and sales forecasting newspaper based on the output of the ARIMA models.

Kaya et al. recommended an 8-layer Deep Neural Network model for vehicle sales prediction. The inputs of the model consist of features, such as exchange rate, the gross domestic product, consumer confidence index, and the consumer price index. The vehicle sales forecast was made according to the output of the model. This work analyzed a total of 90 data monthly between the years of 2011 and 2018 was collected.

Haffner et al. devoted to the sales prediction of Svijany Slovakia, Ltd. Using modern cloud computing tool Microsoft Machine Learning Studio. In the paper there is briefly described characteristics of the company and sales, described the software system Helios Green which is used in the company. There is described the process of data processing from this system for further using. After data processing, predictive models and predictions of the sales are made.

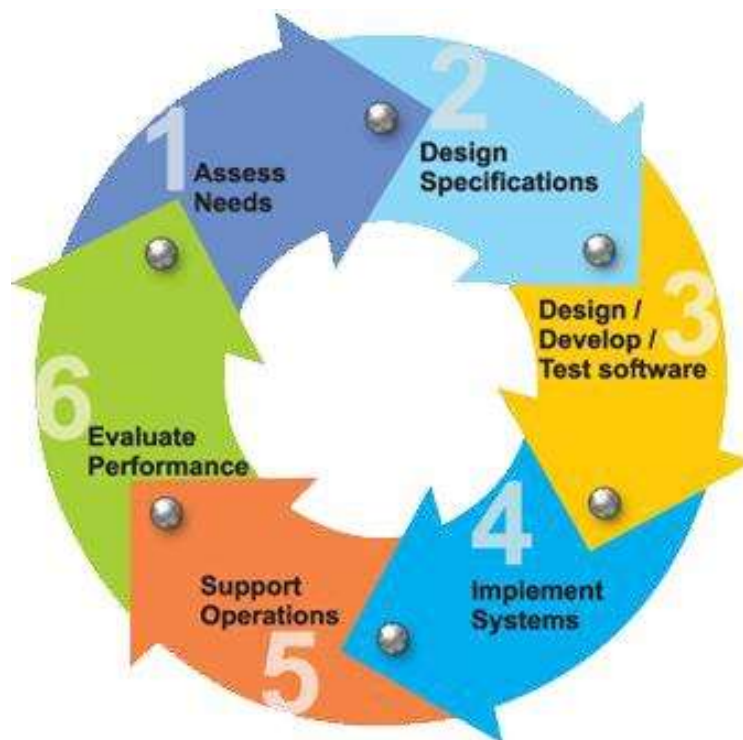
Results of the predications played a significant role in the management negotiations for the building of a new warehouse.

Zhang et al. introduced the seasonal decomposition and ARIMA model, this project proposed a sales forecasting model for the consumer goods with holiday effects. First, a dummy variable model is constructed to test the holiday effects in consumer goods market. Second, using the seasonal decomposition, the seasonal factor is separated from the original series, and the seasonally adjusted series is then obtained. Through the ARIMA model, a trend forecast to the seasonally adjusted series is further carried out. Finally, according to the multiplicative model, refilling the trend forecast value with the seasonal factor, thus, the final sales forecast results of the consumer goods with holiday effects can be obtained. Taking the cigarettes sales in G City, Guizhou, China as an example, the feasibility and effectiveness of this new model is verified by the example analysis results.

### 3. SYSTEM ANALYSIS

#### 3.1 SOFTWARE DEVELOPMENT LIFE CYCLE

There are various software development approaches defined and designed which are used/employed during development process of software, these approaches are also referred as "Software Development Process Models". Each process model follows a particular life cycle in order to ensure success in process of software development.



**Figure 3.1 SOFTWARE DEVELOPMENT LIFE CYCLE**

##### 3.1.1 REQUIREMENTS

Business requirements are gathered in this phase. This phase is the main focus of the project managers and stake holders. Meetings with managers, stake holders and users are held in order to determine the requirements. Who is going to use the system? How will they use the system? What data should be input into the system? What data should be output by the system? These are general questions that get answered during a requirements gathering phase. This produces a nice big list of functionality that the system should provide, which describes functions the system should perform, business logic that processes data, what data is stored and used by the

system, and how the user interface should work. The overall result is the system as a whole and how it performs, not how it is actually going to do it.

### **3.1.2 DESIGN**

The software system design is produced from the results of the requirements phase. Architects have the ball in their court during this phase and this is the phase in which their focus lies. This is where the details on how the system will work is produced. Architecture, including hardware and software, communication, software design (UML is produced here) are all part of the deliverables of a design phase.

### **3.1.3 IMPLEMENTATION**

Code is produced from the deliverables of the design phase during implementation, and this is the longest phase of the software development life cycle. For a developer, this is the main focus of the life cycle because this is where the code is produced. Implementation may overlap with both the design and testing phases. Many tools exist (CASE tools) to actually automate the production of code using information gathered and produced during the design phase.

### **3.1.4 TESTING**

During testing, the implementation is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. Unit tests and system/acceptance tests are done during this phase. Unit tests act on a specific component of the system, while system tests act on the system as a whole. So in a nutshell, that is a very basic overview of the general software development life cycle model. Now let's delve into some of the traditional and widely used variations.

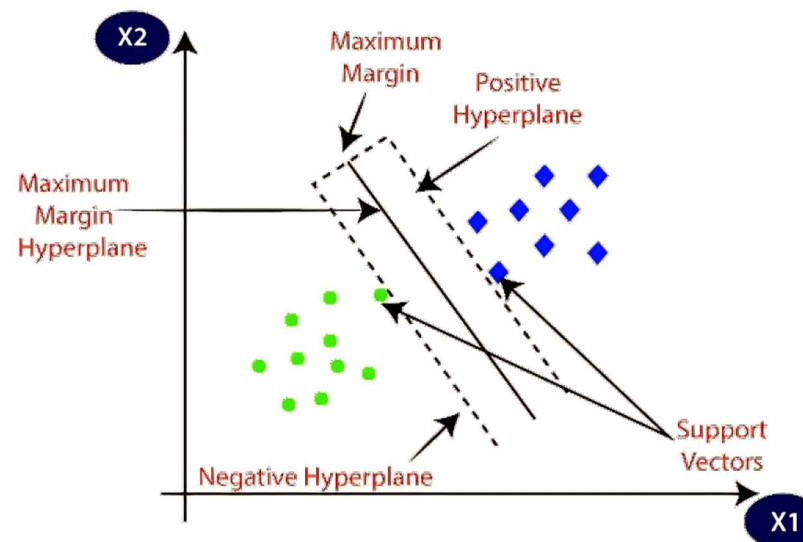
## **3.2 EXISTING SYSTEM**

### **3.2.1 Support Vector Machine Algorithm (SVM)**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that

we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



**Figure 3.2.1 Hyperplane**

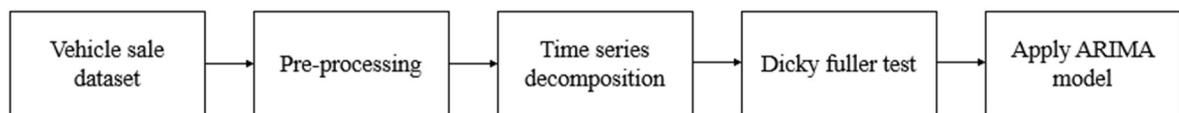
### 3.2.2 Applications

- Face recognition
- Weather prediction
- Medical diagnosis
- Spam detection
- Age/gender identification
- Language identification
- Sentimental analysis
- Authorship identification
- News classification

### 3.2.3 Disadvantages of existing system

- Support vector machine algorithm is not acceptable for large data sets.
- It does not execute very well when the data set has more sound i.e. target classes are overlapping.
- In cases where the number of properties for each data point outstrips the number of training data specimens, the support vector machine will underperform.
- As the support vector classifier works by placing data points, above and below the classifying hyperplane there is no probabilistic clarification for the classification.

## 3.3 PROPOSED SYSTEM



**Fig. 3.3 Block diagram of proposed system.**

### 3.3.1 ARIMA: (Auto Regressive Integrated Moving Average)

ARIMA model is done on time series data. Time series data is a sequence of observations collected from a process with equally spaced periods of time.

Examples: Industrial Averages, Daily data on sales, Daily Customers.

Stages of Time series model process using ARIMA:

ARIMA is also known as Box-Jenkins approach. To build a time series model issuing ARIMA, we need to study the time series and identify  $p, d, q$ . Where,

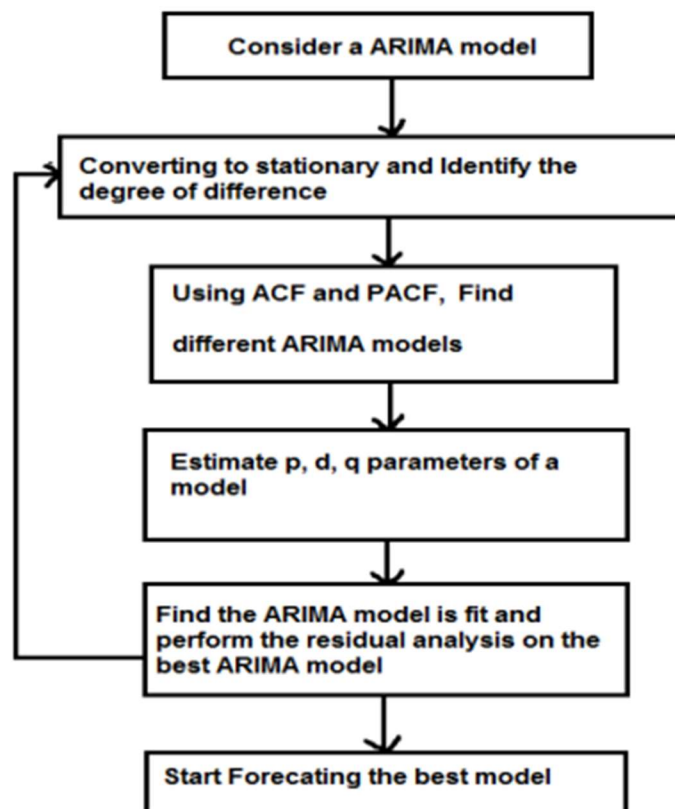
$p$  – Auto Regressive (Auto Correlation)

$d$  - Integrated (Stationary / Trend)

$q$  - Moving Average (Shocks / Error)

- Identification: Determine the appropriate values of  $p, d$ , &  $q$  using the ACF, PACF, and unit root tests.  $p$  is the AR order,  $d$  is the integration order,  $q$  is the MA order

- Estimation: Estimate an ARIMA model using values of p, d, & q you think are appropriate.
- Diagnostic checking: Check residuals of estimated ARIMA model(s) to see if they are white noise; pick best model with well-behaved residuals.
- Forecasting: Produce out of sample forecasts or set aside last few data points for in sample forecasting.



**Fig. 3.3.1 Flow diagram of ARIMA model.**

### **Autoregressive (AR) Process**

Series of current values depend on its own previous values. In an AR(p) model the future value of a variable is assumed to be a linear combination of p past observations and a random error together with a constant term. Mathematically the AR(p) model can be expressed as:

$$y_t = c + \sum_{i=1}^p \varphi_i y_{t-i} + \varepsilon_t = c + \varphi_1 y_{t-1} + \varphi_2 y_{t-2} + \dots + \varphi_p y_{t-p} + \varepsilon_t$$



Here  $y_t$  and  $\varepsilon_t$  are respectively the actual value and random error (random shocks) at period  $t$ ,  $\phi_i$  ( $i = 1, 2, \dots, p$ ) are model parameters and  $c$  is a constant. The integer constant  $p$  is known as the order of the model.

### Moving Average (MA) Process

$$y_t = \mu + \sum_{j=1}^q \theta_j \varepsilon_{t-j} + \varepsilon_t = \mu + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

The current deviation from mean depends on previous deviations. An AR( $p$ ) model regresses against past values of the series, an MA( $q$ ) model uses past errors as the explanatory variables. The MA( $q$ ) model is given by:

$$y_t = \mu + \sum_{j=1}^q \theta_j \varepsilon_{t-j} + \varepsilon_t = \mu + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

Here  $\mu$  is the mean of the series, ( $j=1, 2, \dots, q$ )  $\theta_j$  are the model parameters and  $q$  are the order of the model.

Conceptually a moving average model is a linear regression of the current observation of the time series against the random shocks of one or more prior observations.

### Stationary Series (Integrated)

To model a time series with the Box-Jenkins approach, the series must be stationary. In practical terms, the series is stationary if it tends to wander uniformly about some fixed level. In statistical terms, a stationary process is assumed to be in a particular state of statistical equilibrium, *i.e.*,  $p(x_t)$  is the same for all  $t$ .

To achieve a series as stationary we need to do Regular differencing (RD),

$$\begin{aligned} \text{(1st order)} \quad \nabla x_t &= (1 - B)x_t = x_t - x_{t-1} \\ \text{(2nd order)} \quad \nabla^2 x_t &= (1 - B)^2 x_t = x_t - 2x_{t-1} + x_{t-2} \end{aligned}$$

“B” is the backward shift operator. It is unlikely that more than two regular differencing would ever be needed.

### 3.3.2 Decomposition of time series

The decomposition of time series is a statistical task that deconstructs a time series into several components, each representing one of the underlying categories of patterns. There are two principal types of decomposition, which are outlined below.

#### Decomposition based on rates of change

This is an important technique for all types of time series analysis, especially for seasonal adjustment. It seeks to construct, from an observed time series, a number of component series (that could be used to reconstruct the original by additions or multiplications) where each of these has a certain characteristic or type of behavior. For example, time series are usually decomposed into:

- $T_t$ , the trend component at time  $t$ , which reflects the long-term progression of the series (secular variation). A trend exists when there is a persistent increasing or decreasing direction in the data. The trend component does not have to be linear.
- $C_t$ , the cyclical component at time  $t$ , which reflects repeated but non-periodic fluctuations. The duration of these fluctuations depends on the nature of the time series.
- $S_t$ , the seasonal component at time  $t$ , reflecting seasonality (seasonal variation). A seasonal pattern exists when a time series is influenced by seasonal factors. Seasonality occurs over a fixed and known period (e.g., the quarter of the year, the month, or day of the week).
- $I_t$  the irregular component (or "noise") at time  $t$ , which describes random, irregular influences. It represents the residuals or remainder of the time series after the other components have been removed.

Hence a time series using an additive model can be thought of as

$$y_t = T_t + C_t + S_t + I_t$$

whereas a multiplicative model would be

$$y_t = T_t \times C_t \times S_t \times I_t$$

An additive model would be used when the variations around the trend do not vary with the level of the time series whereas a multiplicative model would be appropriate if the trend is proportional to the level of the time series.

Sometimes the trend and cyclical components are grouped into one, called the trend-cycle component. The trend-cycle component can just be referred to as the "trend" component, even though it may contain cyclical behavior. For example, a seasonal decomposition of time series by Loess (STL) plot decomposes a time series into seasonal, trend and irregular components using loess and plots the components separately, whereby the cyclical component (if present in the data) is included in the "trend" component plot.

### **Decomposition based on predictability**

The theory of time series analysis makes use of the idea of decomposing a times series into deterministic and non-deterministic components (or predictable and unpredictable components).

### **3.3.3 Pre-processing**

#### ***Data Pre-processing in Machine learning***

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So, for this, we use data pre-processing task.

#### ***Why do we need Data Pre-processing?***

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data pre-processing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

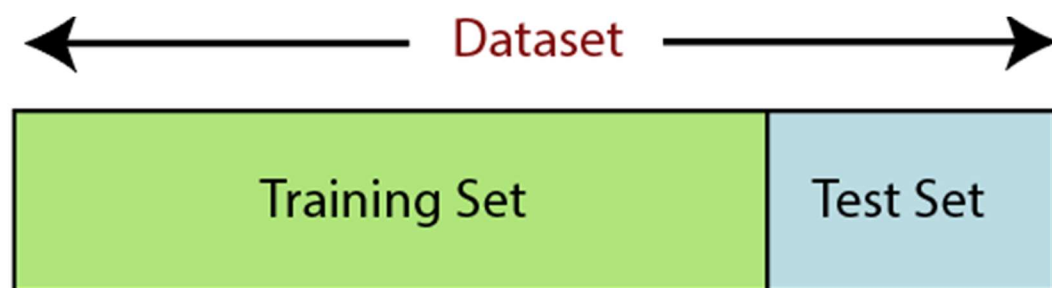
- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

### 3.3.4 Splitting the Dataset into the Training set and Test set

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model.

Suppose if, we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



**Figure 3.2.4 Dataset**

**Training Set:** A subset of dataset to train the machine learning model, and we already know the output.

**Test set:** A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

### 3.2.5 ADVANTAGES OF ARIMA

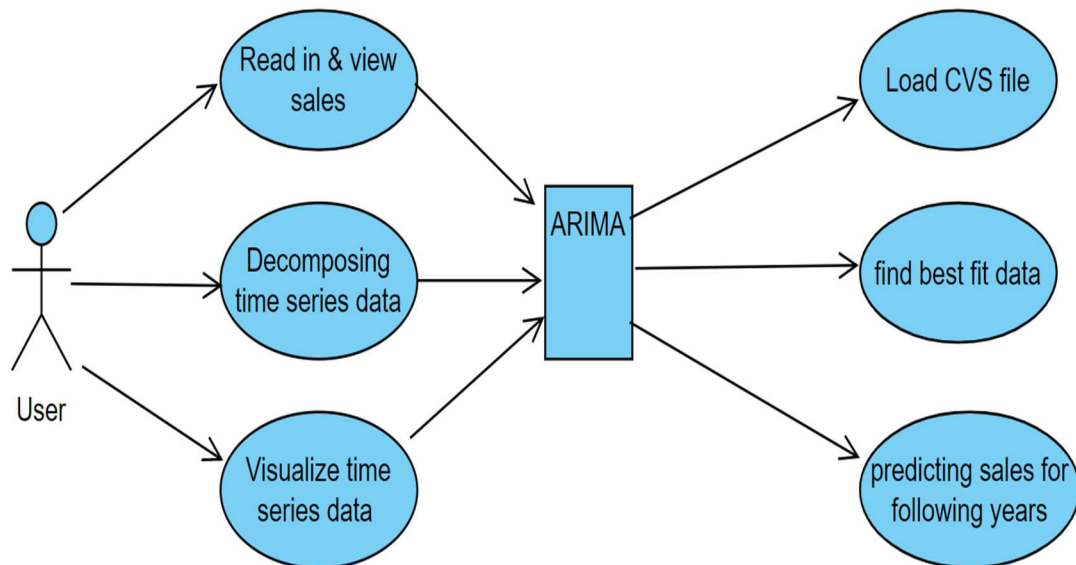
- It's accuracy of predicting future vehicle sales in interval of 95% to 99%.
- It's produces reliable results then other models.

## 4. SYSTEM DESIGN

### 4.1 UML DAIGRAMS

UML combines best techniques from data modeling (entity relationship diagrams), business modeling (work flows), object modeling, and component modeling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies. UML has synthesized the notations of the Booch method, the Object-modeling technique (OMT) and Object-oriented software engineering (OOSE) by fusing them into a single, common and widely usable modeling language. UML aims to be a standard modeling language which can model concurrent and distributed systems.

#### 4.1.1 Use Case



**Figure 4.1.1 Use Case Diagram**

### 4.1.2 Class diagram

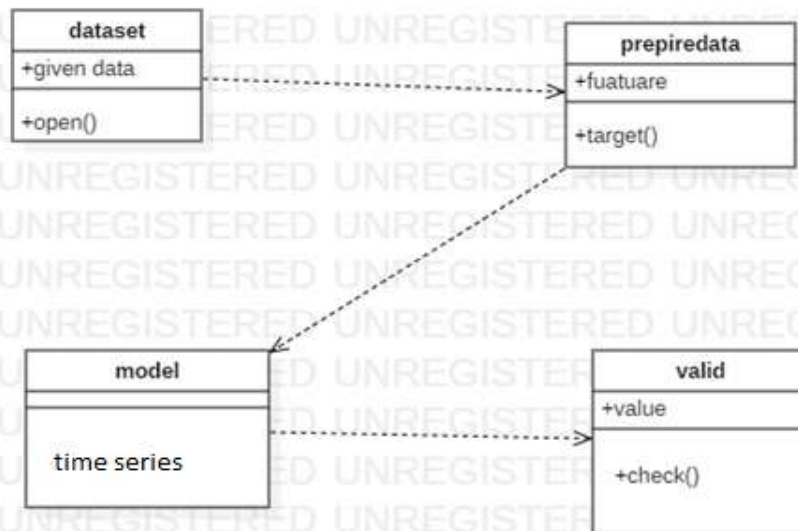


Figure 4.1.2 Class Diagram

### 4.1.3 Activity diagram

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

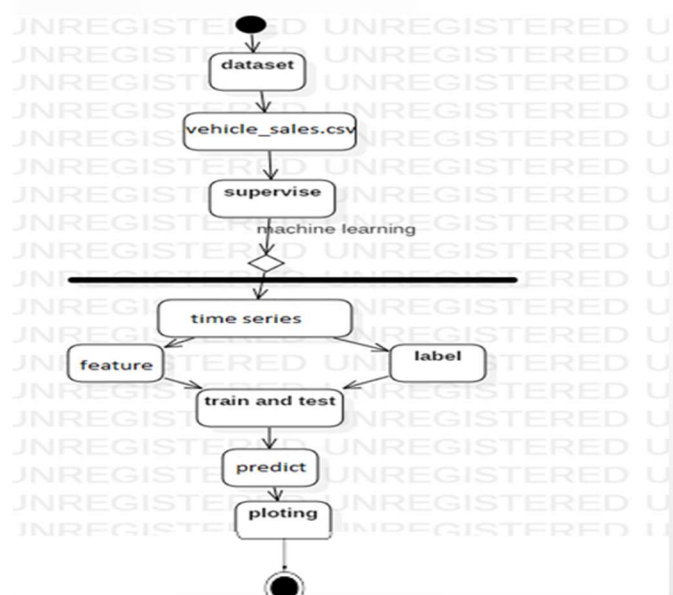


Figure 4.1.3 Activity Diagram

#### 4.1.4 Sequence diagram

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows, as parallel vertical lines ("lifelines"), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

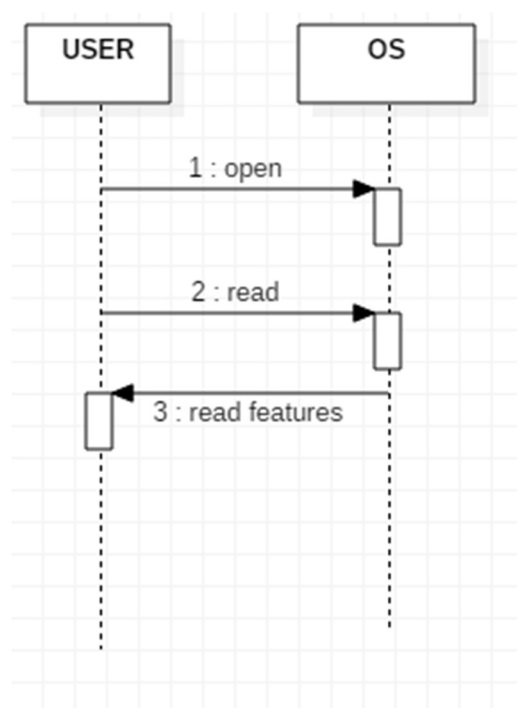
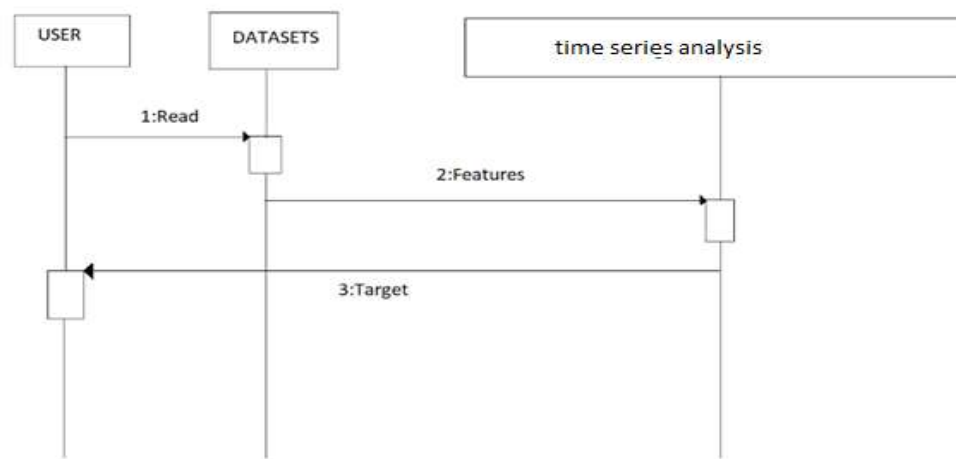


Figure 4.1.4 Sequence Diagram

#### 4.1.5 Component diagram

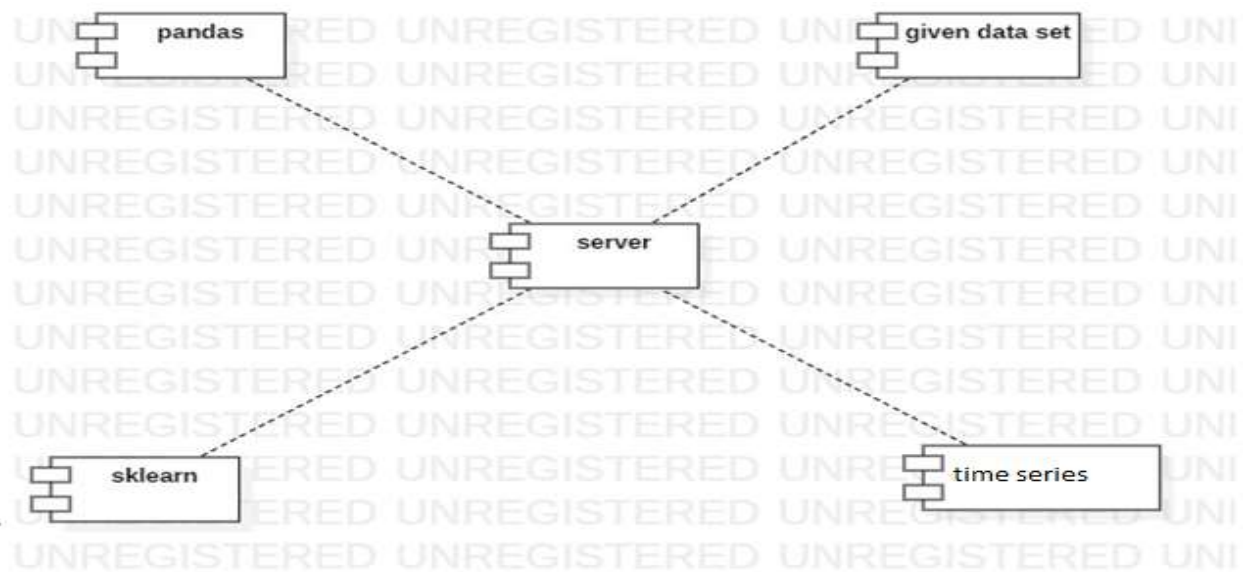


Figure 4.1.5 Component Diagram

#### 4.1.6 Interaction Overview Diagram

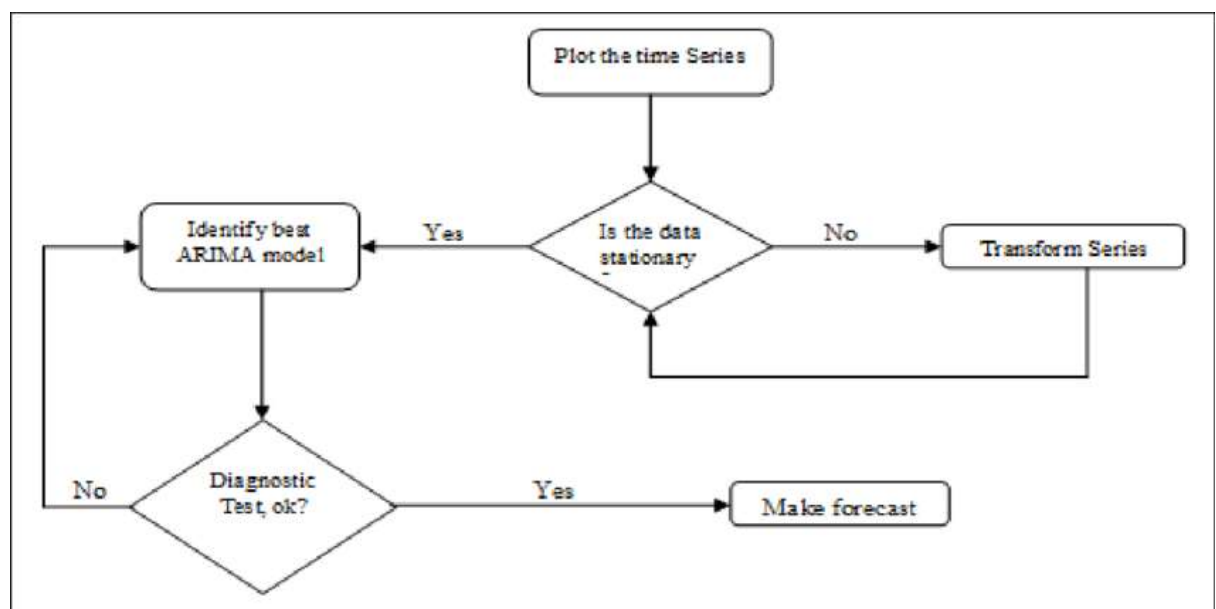


Figure 4.1.6 Interaction Overview Diagram



## **5. SOFTWARE ENVIRONMENT**

### **5.1 MACHINE LEARNING**

#### **What is Machine Learning**

Before we look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

#### **Categories of Machine Learning**

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label and is often described as "letting the dataset speak for itself." These models include tasks such as clustering and dimensionality reduction. Clustering algorithms identify distinct groups

of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

### **Need for Machine Learning**

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate, and solve complex problems. On the other side, AI is still in its initial stage and have not surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programming logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

### **Challenges in Machines Learning**

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are –

1. Quality of data – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.
2. Time-Consuming task – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.
3. Lack of specialist persons – As ML technology is still in its infancy stage, availability of expert resources is a tough job.
4. No clear objective for formulating business problems – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology

is not that mature yet.

5. Issue of overfitting & underfitting – If the model is overfitting or underfitting, it cannot be represented well for the problem.
6. Curse of dimensionality – Another challenge ML model faces is too many features of data points. This can be a real hindrance.
7. Difficulty in deployment – Complexity of the ML model makes it quite difficult to be deployed in real life.

### **Applications of Machines Learning**

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

### **How to Start Learning Machine Learning?**

Arthur Samuel coined the term “Machine Learning” in 1959 and defined it as a “Field of study that gives computers the capability to learn without being explicitly programmed”.

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is the Best Job of 2019 with a 344% growth and an average base salary of \$146,085 per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So, this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let’s get started!!!

### **How to start learning ML?**

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

#### **Step 1 – Understand the Prerequisites**

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don’t know these, never fear! You don’t need a Ph.D. degree in these topics to get started but you do need a basic understanding.

##### **(a) Learn Linear Algebra and Multivariate Calculus**

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

##### **(b) Learn Statistics**

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!!

Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

### (c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as Fork Python available Free on GeeksforGeeks.

### Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

#### (a) Terminologies of Machine Learning

- **Model** – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature** – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the

categories trained on.

- Prediction – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

### **(b) Types of Machine Learning**

- Supervised Learning – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- Unsupervised Learning – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- Semi-supervised Learning – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- Reinforcement Learning – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

### **Advantages of Machine learning**

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

### 3. Continuous Improvement

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

### 4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

### 5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

## **Disadvantages of Machine Learning**

### 1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

### 2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

### 3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

### 4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to

customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

## **5.2 PYTHON**

### **What is Python?**

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.

The biggest strength of Python is huge collection of standard libraries which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc. )
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

### **Advantages of Python**

Let's see how Python dominates over other languages.

#### **1. Extensive Libraries**

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI,



email, image manipulation, and more. So, we don't have to write the complete code for that manually.

## 2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

## 3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

## 4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

## 5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

## 6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

## 7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. This further aids the readability of the code.

## 8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

## 9. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

## 10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

## 11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

## Advantages of Python Over Other Languages

### 1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

### 2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

### 3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

### **Disadvantages of Python**

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

#### 1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

#### 2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

#### 3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

#### 4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

## 5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

## History of Python

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners<sup>1</sup>, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

## Python Development Steps

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system.

Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it." Some changes in Python 3:

Print is now a function.

- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g., a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e., int. long is int as well.
- The division of two integers returns a float instead of an integer. "/" can be used to have the "old" behaviour.
- Text Vs. Data Instead of Unicode Vs. 8-bit

## **Purpose**

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

Install Python Step-by-Step in Windows and Mac

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

### How to Install Python on Windows and Mac

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet [here](#). The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

### Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e., Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

### **Installation of Python**

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.

Step 2: Before you click on Install Now, make sure to put a tick on Add Python 3.7 to PATH.

Step 3: Click on Install NOW After the installation is successful. Click on Close.

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

### **Verify the Python Installation**

Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.

Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python -V and press Enter.

Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. enter print (“Hey World”) and Press Enter.

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won't work.



## **6. SYSTEM REQUIREMENTS SPECIFICATION**

### **6.1 SYSTEM REQUIREMENTS**

#### **6.1.1 SOFTWARE REQUIREMENTS**

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

- Python IDLE 3.7
- Anaconda 3.7
- Jupiter

#### **6.1.2 HARDWARE REQUIREMENTS**

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

- Operating system : Windows
- Processor : intel i5
- Ram : 8 GB
- Hard disk : 512 GB

### **6.2 FUNCTIONAL REQUIREMENTS**

#### **6.2.1 OUTPUT DESIGN**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

## **OUTPUT DEFINITION**

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

### **6.2.2 INPUT DESIGN**

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

## **INPUT STAGES**

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification

- Data control
- Data transmission
- Data validation
- Data correction

## **INPUT TYPES**

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

## **INPUT MEDIA**

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

## **ERROR AVOIDANCE**

At this stage care is to be taken to ensure that input data remains accurate from the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

## **ERROR DETECTION**

Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

## **DATA VALIDATION**

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

## **USER INTERFACE DESIGN**

It is essential to consult the system users and discuss their needs while designing the user interface:

### **USER INTERFACE SYSTEMS CAN BE BROADLY CLASSIFIED AS:**

- User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
- Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

## **USER INITIATED INTERFACES**

User initiated interfaces fall into two approximate classes:

- Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
- Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice.

## **COMPUTER-INITIATED INTERFACES**

The following computer – initiated interfaces were used:

- The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
- Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

## **ERROR MESSAGE DESIGN**

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

## **PERFORMANCE REQUIREMENTS**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system
- The system should be accurate
- The system should be better than the existing system
- The existing system is completely dependent on the user to perform all the duties.

## 7. SYSTEM IMPLEMENTATIONS

### 7.1 Modules Used in Project

#### TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

#### NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

#### Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with

Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

## **Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc., via an object-oriented interface or via a set of functions familiar to MATLAB users.

## **Scikit – learn**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

## **7.2 SAMPLE CODE**

```
import warnings
import itertools
import pandas as pd
import numpy as np
import statsmodels.api as sm
import statsmodels.tsa.api as smt
import statsmodels.formula.api as smf
import matplotlib.pyplot as plt

plt.style.use('bmh')
# Visualize Tractor Sales data as time series
```



```

from io import StringIO
import requests
import os
print(os.getcwd())

os.chdir('E:\\mini project\\14. SALES OF VEHICLES BY USING TIMESERIES')

sales_data = pd.read_csv('PH-Sales.csv')
sales_data.tail(10)

sales_data.shape

# since the complete date was not mentioned, we assume that it was the first of every month
dates = pd.date_range(start='2003-01-01', freq='MS', periods=len(sales_data))
dates

import calendar
sales_data['Month'] = dates.month
sales_data.head(5)

sales_data['Month'] = sales_data['Month'].apply(lambda x: calendar.month_abbrev[x])
sales_data.head(5)

sales_data['Year'] = dates.year
sales_data.head(5)

sales_data.drop(['Month-Year'], axis=1, inplace=True)
sales_data.head(5)

sales_data.rename(columns={'Number of Tractor Sold': 'Tractor-Sales'}, inplace=True)
sales_data.head(5)

sales_data = sales_data[['Month', 'Year', 'Tractor-Sales']]
sales_data.head(5)

# set the dates as the index of the dataframe, so that it can be treated as a
# time-series dataframe
sales_data.set_index(dates, inplace=True)

```

```

sales_data.head(5)

# extract out the time-series
sales_ts = sales_data['Tractor-Sales']
plt.figure(figsize=(10, 5))
plt.plot(sales_data['Tractor-Sales'])
plt.title("trend or sea")
plt.xlabel('Years')
plt.ylabel('Tractor Sales')

# PH Trend - Time Series Decomposition
# remove wrinkles from our time series using moving average.
# moving average of different time periods i.e. 4,6,8, and 12 months
fig, axes = plt.subplots(2, 2, sharey=False, sharex=False)
fig.set_figwidth(14)
fig.set_figheight(8)
axes[0][0].plot(sales_ts.index, sales_ts, label='Original')
axes[0][0].plot(sales_ts.index, sales_ts.rolling(window=4).mean(), label='4-Months Rolling
Mean')
axes[0][0].set_xlabel("Years")
axes[0][0].set_ylabel("Number of Tractor's Sold")
axes[0][0].set_title("4-Months Moving Average")
axes[0][0].legend(loc='best')

axes[0][1].plot(sales_ts.index, sales_ts, label='Original')
axes[0][1].plot(sales_ts.index, sales_ts.rolling(window=6).mean(), label='6-Months Rolling
Mean')
axes[0][1].set_xlabel("Years")
axes[0][1].set_ylabel("Number of Tractor's Sold")
axes[0][1].set_title("6-Months Moving Average")
axes[0][1].legend(loc='best')

axes[1][0].plot(sales_ts.index, sales_ts, label='Original')
axes[1][0].plot(sales_ts.index, sales_ts.rolling(window=8).mean(), label='8-Months Rolling
Mean')
axes[1][0].set_xlabel("Years")
axes[1][0].set_ylabel("Number of Tractor's Sold")

```

```

axes[1][0].set_title("8-Months Moving Average")
axes[1][0].legend(loc='best')

axes[1][1].plot(sales_ts.index, sales_ts, label='Original')
axes[1][1].plot(sales_ts.index, sales_ts.rolling(window=12).mean(), label='12-Months Rolling
Mean')
axes[1][1].set_xlabel("Years")
axes[1][1].set_ylabel("Number of Tractor's Sold")
axes[1][1].set_title("12-Months Moving Average")
axes[1][1].legend(loc='best')
plt.tight_layout()
plt.show()

```

```

"""# 12-month moving average produces a wrinkle free curve.
# There is expected monthly-seasonal effect in our data."""
# plot the rolling mean and standard deviation on window of 12 months.
# Determining rolling statistics
rolmean = sales_ts.rolling(window=12).mean()
rolstd = sales_ts.rolling(window=12).std()
#Plot rolling statistics:
orig = plt.plot(sales_ts, label='Original')
mean = plt.plot(rolmean, label='Rolling Mean')
std = plt.plot(rolstd, label = 'Rolling Std')
plt.legend(loc='best')
plt.title('Rolling Mean & Standard Deviation')
plt.show(block=False)

```

```

# PH Tractor - Dicky Fuller Test on the timeseries
# run the Dicky Fuller Test on the timeseries and
# verify the null hypothesis that the TS is non-stationary.
# Perform Dickey-Fuller test:
from statsmodels.tsa.stattools import adfuller
print('Results of Dickey-Fuller Test:')
dfctest = adfuller(sales_ts, autolag='AIC')
dfcoutput = pd.Series(dfctest[0:4], index=['Test Statistic', 'p-value', '#lags Used', 'Number of
Observations Used'])
for key, value in dfctest[4].items():

```

```

dfoutput['Critical Value (%s)%key] = value
print(dfoutput)

"""# Though the variation in standard deviation is small,
# rolling mean is clearly increasing with time and
# this is not a stationary series.
# we observe the moving average over months that there is a monhly pattern,
"""

# PH Tractor Seasonality – Time Series Decomposition
# The first thing to do is to see how number of tractors sold vary on a
# month on month basis. plot a stacked annual plot to observe seasonality
sales_data.head()

monthly_sales_data = pd.pivot_table(sales_data, values = "Tractor-Sales", columns = "Year",
index = "Month")
monthly_sales_data

monthly_sales_data = monthly_sales_data.reindex(index = ['Jan','Feb','Mar', 'Apr', 'May',
'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])
monthly_sales_data

monthly_sales_data.plot()
yearly_sales_data = pd.pivot_table(sales_data, values = "Tractor-Sales", columns = "Month",
index = "Year")
yearly_sales_data = yearly_sales_data[['Jan','Feb','Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep',
'Oct', 'Nov', 'Dec']]
yearly_sales_data

yearly_sales_data.boxplot()

"""# Important Inferences
# The tractor sales have been increasing without fail every year.
# July and August are the peak months for tractor sales and the variance and
# the mean value in July and August are also much higher than any of the
# other months.
# We can see a seasonal cycle of 12 months where the mean value of each month
# starts with a increasing trend in the beginning of the year and drops down
# towards the end of the year. We can see a seasonal effect with a cycle of

```

```

# 12 months."
# PH Tractor Irregular Remainder
decomposition = sm.tsa.seasonal_decompose(sales_ts, model='multiplicative')
fig = decomposition.plot()
fig.set_figwidth(12)
fig.set_figheight(8)
fig.suptitle('Decomposition of multiplicative time series')
plt.show()

# Step 2: Difference data to make data stationary on mean (remove trend)
plt.figure(figsize=(10, 5))
plt.plot(sales_ts.diff(periods=1))
plt.xlabel('Years')
plt.ylabel('Tractor Sales')

# original
plt.figure(figsize=(10, 5))
plt.plot(sales_ts)
plt.xlabel('Years')
plt.ylabel('Tractor Sales')

# Step 3: log transform data to make data stationary on variance
plt.figure(figsize=(10, 5))
plt.plot(np.log10(sales_ts))
plt.xlabel('Years')
plt.ylabel('Log (Tractor Sales)')

# Step 4: Difference log transform data to make data stationary on both mean and
# variance
plt.figure(figsize=(10, 5))
plt.plot(np.log10(sales_ts).diff(periods=1))
plt.xlabel('Years')
plt.ylabel('Differenced Log (Tractor Sales)')

# Step 5: Plot ACF and PACF to identify potential AR and MA model
sales_ts_log = np.log10(sales_ts)
sales_ts_log.dropna(inplace=True)

```

```

sales_ts_log.head(5)

sales_ts_log_diff = sales_ts_log.diff(periods=1) # same as ts_log_diff = ts_log -
ts_log.shift(periods=1)
sales_ts_log_diff.dropna(inplace=True)
fig, axes = plt.subplots(1, 2, sharey=False, sharex=False)
fig.set_figwidth(12)
fig.set_figheight(4)
smt.graphics.plot_acf(sales_ts_log_diff, lags=30, ax=axes[0], alpha=0.5)
smt.graphics.plot_pacf(sales_ts_log_diff, lags=30, ax=axes[1], alpha=0.5)
plt.tight_layout()

#ACF and PACF plots:
from statsmodels.tsa.stattools import acf, pacf
lag_acf = acf(sales_ts_log_diff, nlags=20)
lag_pacf = pacf(sales_ts_log_diff, nlags=20, method='ols')

plt.figure(figsize=(10,8))

#Plot ACF:
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(sales_ts_log_diff)),linestyle='--',color='gray')
plt.axhline(y=1.96/np.sqrt(len(sales_ts_log_diff)),linestyle='--',color='gray')
plt.title('Autocorrelation Function')

#Plot PACF:
plt.subplot(122)
plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(sales_ts_log_diff)),linestyle='--',color='gray')
plt.axhline(y=1.96/np.sqrt(len(sales_ts_log_diff)),linestyle='--',color='gray')
plt.title('Partial Autocorrelation Function')
plt.tight_layout()

# Step 6: Identification of best fit ARIMA model

```

```

# Define the p, d and q parameters to take any value between 0 and 2
p = d = q = range(0, 2)
# Generate all different combinations of p, d and q triplets
pdq = list(itertools.product(p, d, q))
pdq

# Generate all different combinations of seasonal p, q and q triplets
seasonal_pdq = [(x[0], x[1], x[2], 12) for x in list(itertools.product(p, d, q))]
seasonal_pdq

import sys
warnings.filterwarnings("ignore") # specify to ignore warning messages
best_aic = np.inf
best_pdq = None
best_seasonal_pdq = None
temp_model = None
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            temp_model = sm.tsa.statespace.SARIMAX(sales_ts_log, order =
                                                    param, seasonal_order = param_seasonal,
                                                    enforce_stationarity=True,
                                                    enforce_invertibility=True)

            results = temp_model.fit()
            # print("SARIMAX{x}{x}12 - AIC: {}".format(param, param_seasonal, results.aic))
            if results.aic < best_aic:
                best_aic = results.aic
                best_pdq = param
                best_seasonal_pdq = param_seasonal
        except:
            #print("Unexpected error:", sys.exc_info()[0])
            continue
print("Best SARIMAX{x}{x}12 model - AIC: {}".format(best_pdq, best_seasonal_pdq,
best_aic))

#AIC & BIC

```

```

blog_param_order = (0, 1, 1)
blog_param_seasonal_order = (0, 1, 1, 12)
blog_model = sm.tsa.statespace.SARIMAX(sales_ts_log,
order=blog_param_order,
seasonal_order=blog_param_seasonal_order,
enforce_stationarity=True,
enforce_invertibility=True)
blog_results = blog_model.fit()
print("Blog SARIMAX {}x{}12 model - AIC: {}".format(blog_param_order,
blog_param_seasonal_order, blog_results.aic))

# Step 7: Predict sales on in-sample date using the best fit ARIMA model
best_model = sm.tsa.statespace.SARIMAX(sales_ts_log,
order=(0, 1, 1),
seasonal_order=(1, 0, 1, 12),
enforce_stationarity=True,
enforce_invertibility=True)
best_results = best_model.fit()
print(best_results.summary().tables[0])
print(best_results.summary().tables[1])

pred_dynamic = best_results.get_prediction(start=pd.to_datetime('2012-01-01'),
dynamic=True, full_results=True)
pred_dynamic_ci = pred_dynamic.conf_int()
# Extract the predicted and true values of our time series
sales_ts_forecasted = pred_dynamic.predicted_mean
sales_ts_truth = sales_ts_log['2012-01-01:']
sales_ts_truth

# Compute the mean square error
mse = ((sales_ts_forecasted - sales_ts_truth) ** 2).mean()
print('The Mean Squared Error of our forecasts is {}'.format(round(mse, 4)))

# The Mean Squared Error of our forecasts is 0.0011
axis = sales_ts['2005:'].plot(label='Observed', figsize=(10, 6))
np.power(10, pred_dynamic.predicted_mean).plot(ax=axis, label='Dynamic Forecast',
alpha=0.7)

```



```

axis.fill_between(pred_dynamic_ci.index, pred_dynamic_ci.iloc[:, 0], pred_dynamic_ci.iloc[:,
1], color='k', alpha=.25)
axis.fill_betweenx(axis.get_ylim(), pd.to_datetime('2012-01-01'), sales_ts.index[-1], alpha=.1,
zorder=-1)
axis.set_xlabel('Years')
axis.set_ylabel('Tractor Sales')
plt.legend(loc='best')
plt.show()
plt.close()

```

# Step 8: Forecast sales using the best fit ARIMA model

```

n_steps = 36
pred_uc_99 = best_results.get_forecast(steps=36, alpha=0.01) # alpha=0.01 signifies 99%
confidence interval
pred_uc_95 = best_results.get_forecast(steps=36, alpha=0.05) # alpha=0.05 95% CI
# Get confidence intervals 95% & 99% of the forecasts
pred_ci_99 = pred_uc_99.conf_int()
pred_ci_95 = pred_uc_95.conf_int()
n_steps = 36
idx = pd.date_range(sales_ts.index[-1], periods=n_steps, freq='MS')
fc_95 = pd.DataFrame(np.column_stack([np.power(10, pred_uc_95.predicted_mean),
np.power(10, pred_ci_95)]),
index=idx, columns=['forecast', 'lower_ci_95', 'upper_ci_95'])
fc_99 = pd.DataFrame(np.column_stack([np.power(10, pred_ci_99)]),
index=idx, columns=['lower_ci_99', 'upper_ci_99'])
fc_all = fc_95.combine_first(fc_99)
fc_all = fc_all[['forecast', 'lower_ci_95', 'upper_ci_95', 'lower_ci_99', 'upper_ci_99']] # just
reordering columns
fc_all.head()

```

# plot the forecast along with the confidence band

```

axis = sales_ts.plot(label='Observed', figsize=(15, 6))
fc_all['forecast'].plot(ax=axis, label='Forecast', alpha=0.7)
axis.fill_between(fc_all.index, fc_all['lower_ci_95'], fc_all['upper_ci_95'], color='k',
alpha=.25)
#axis.fill_between(fc_all.index, fc_all['lower_ci_99'], fc_all['upper_ci_99'], color='k',
alpha=.75)
axis.set_xlabel('Years')

```

```
axis.set_ylabel('Tractor Sales')  
plt.legend(loc='best')  
plt.show()
```

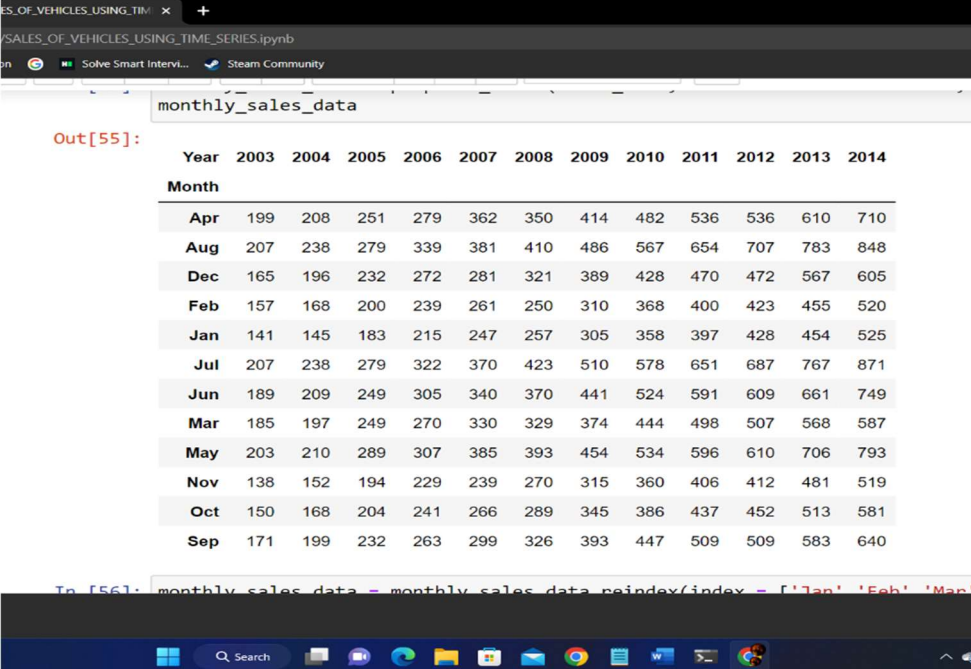
```
# Step 9: Plot ACF and PACF for residuals of ARIMA model to ensure no  
best_results.plot_diagnostics(lags=30, figsize=(16,12))  
plt.show()
```

## 8. SYSTEM RESULT

The performance of ARIMA is checked by plotting the graph for the confusion matrix, which is generated for the temperature against years. A confusion matrix can help visualize the results of a ARIMA classification algorithm. In this project the implementation is done in the programming language R. Here, we can show that results by compare to other models ARIMA can perform better.

### ORIGINAL DATA

Original data that is available in the excel sheet.



Out[55]:

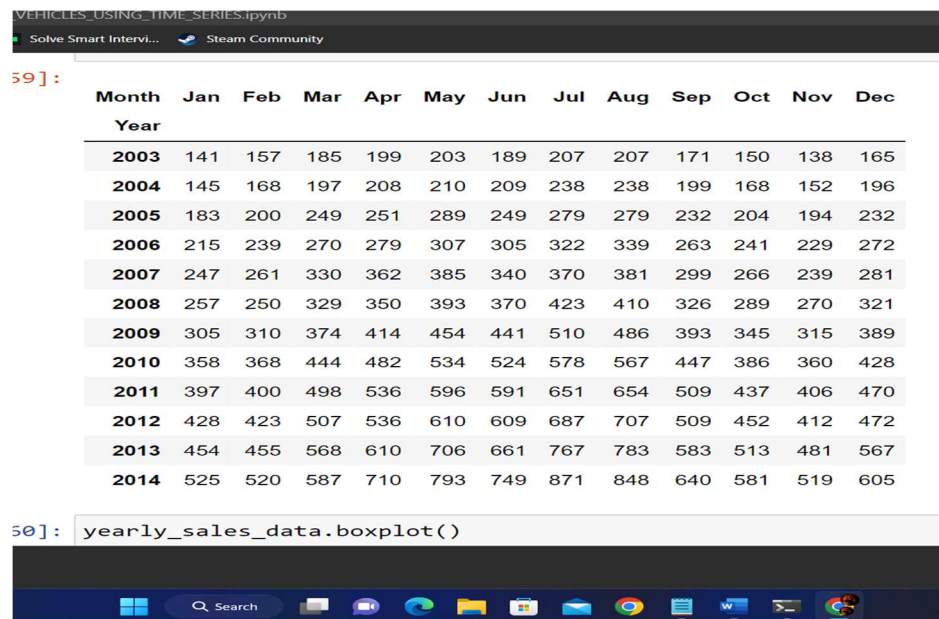
Year	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014
Month												
Apr	199	208	251	279	362	350	414	482	536	536	610	710
Aug	207	238	279	339	381	410	486	567	654	707	783	848
Dec	165	196	232	272	281	321	389	428	470	472	567	605
Feb	157	168	200	239	261	250	310	368	400	423	455	520
Jan	141	145	183	215	247	257	305	358	397	428	454	525
Jul	207	238	279	322	370	423	510	578	651	687	767	871
Jun	189	209	249	305	340	370	441	524	591	609	661	749
Mar	185	197	249	270	330	329	374	444	498	507	568	587
May	203	210	289	307	385	393	454	534	596	610	706	793
Nov	138	152	194	229	239	270	315	360	406	412	481	519
Oct	150	168	204	241	266	289	345	386	437	452	513	581
Sep	171	199	232	263	299	326	393	447	509	509	583	640

In [56]: monthly\_sales\_data = monthly\_sales\_data.reindex(index = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])

SCREENSHOT 8.1 ORIGINAL DATA

### DATA PRE-PROCESSING

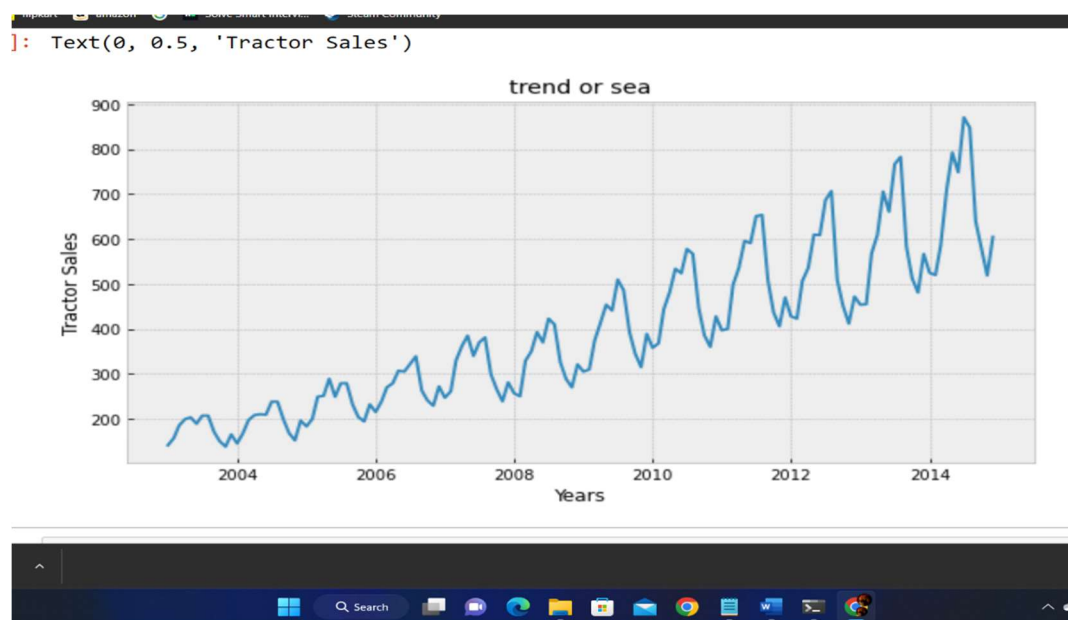
Original data is pre-processed before applying algorithms.



SCREENSHOT 8.2 PRE-PROCESSED DATA

## EXTRACTING TIME SERIES

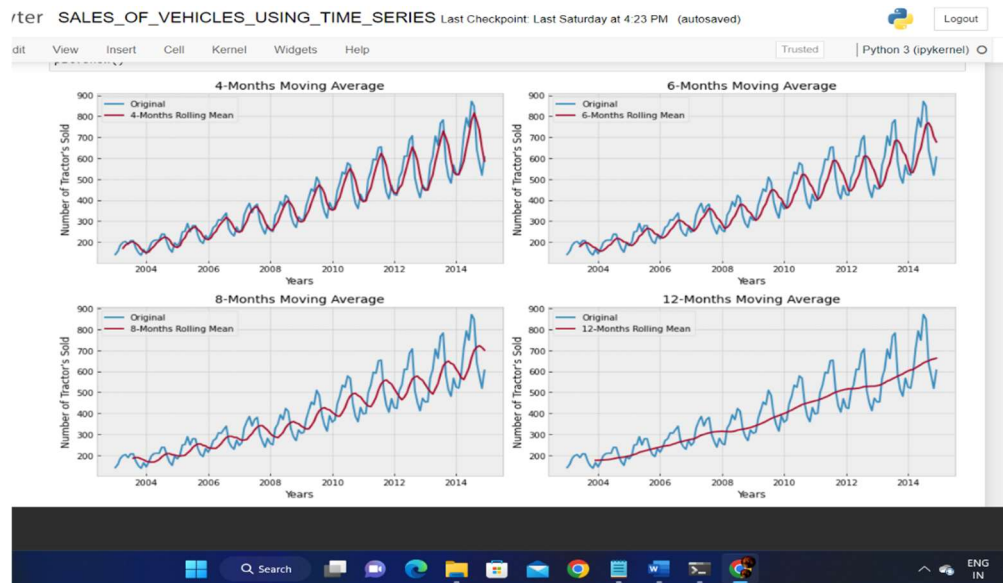
Extracting the time series from the pre-processed data to apply arima.



Screenshot 8.3 Extracting Time Series

## REMOVING WRINKLES

Removing Wrinkles from our time series using moving average.



Screenshot 8.4 Removing Wrinkles

## DICKEY-FULLER TEST

Applying Dickey-Fuller test on our time series to verify the null hypothesis that our timeseries is Non-Stationary.

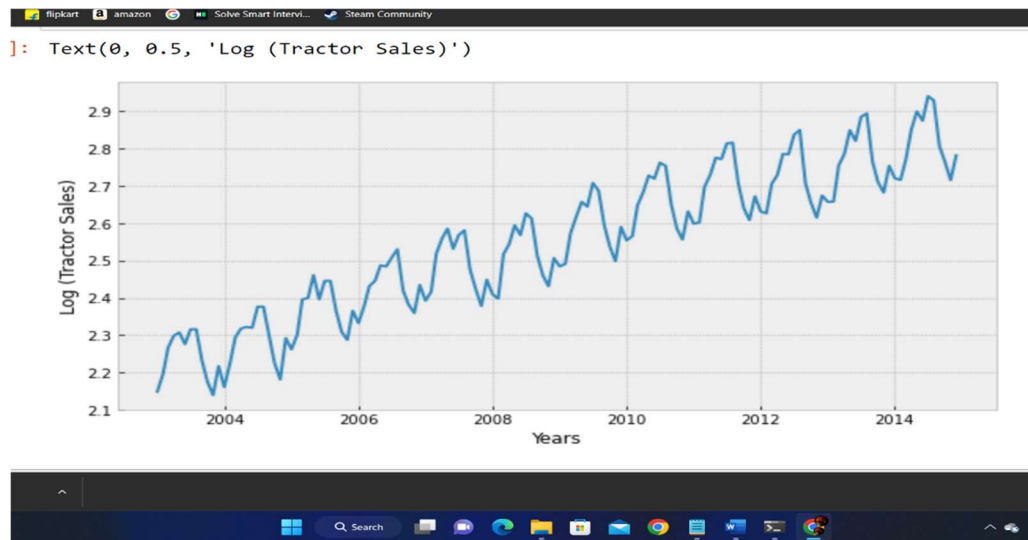
```
Results of Dickey-Fuller Test:
Test Statistic           1.108825
p-value                   0.995291
#lags Used                14.000000
Number of Observations Used 129.000000
Critical Value (1%)       -3.482088
Critical Value (5%)       -2.884219
Critical Value (10%)      -2.578864
dtype: float64
```

```
· '''# Though the variation in standard deviat
```

Screenshot 8.5 Dickey-Fuller Test

## APPLYING LOG

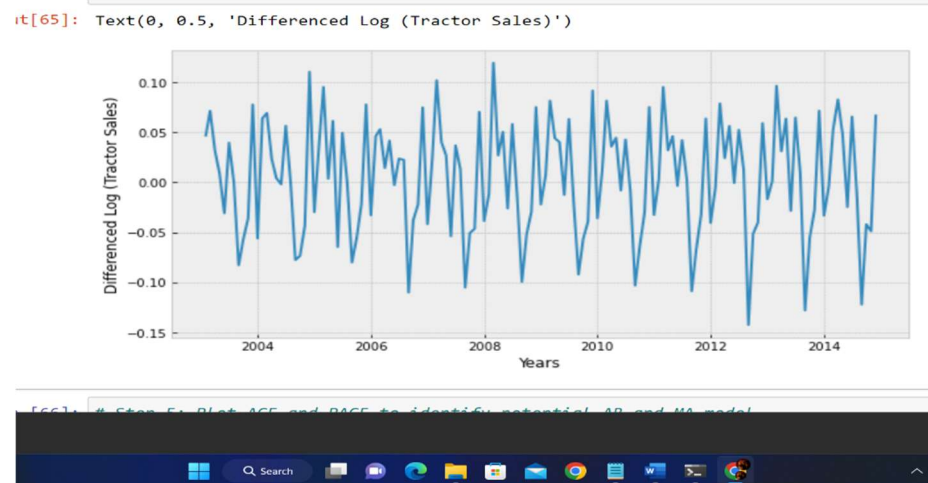
Applying log to make our time series stationary on variance.



Screenshot 8.1.1 Applying Log

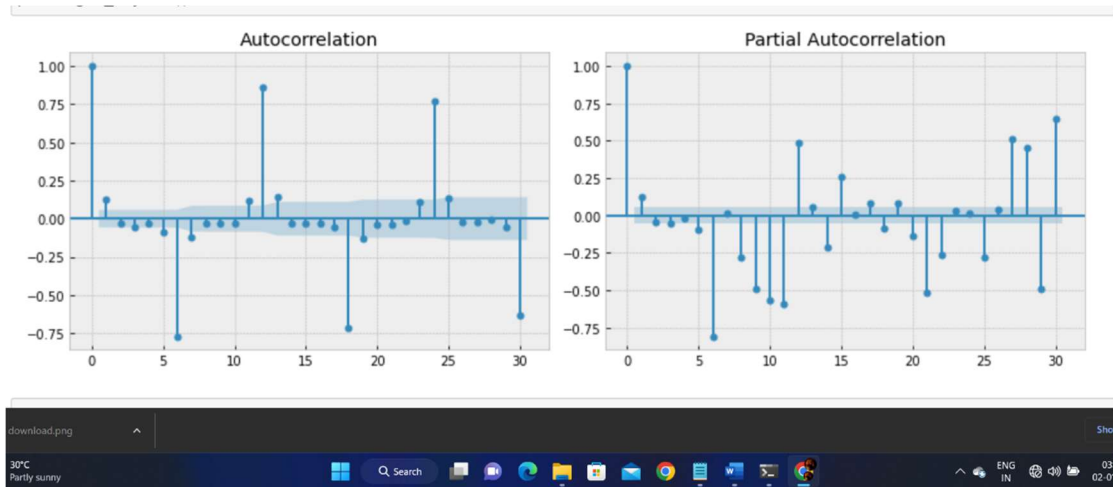
## APPLYING DIFFERENCE LOG

Applying difference log to make our time series stationary on both variance and mean.



Screenshot 8.7 Applying Difference log

## ACF AND PACF OF ORIGINAL DATA



**Screenshot 8.8 ACF and PACF of Original Data**

From the two ACF and PACF graphs we have to find the potential ARIMA fit from which we are going to predict the sales forecasting. When choosing the model which is best fit it should be based on AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion). Normally, the model which has lowest AIC value is taken is best fit.

### Best Fit ARIMA Model

Predict sales on in-sample date using the best fit ARIMA model

```

=====
SARIMAX Results
=====
Dep. Variable:          Tractor-Sales    No. Observations:      144
Model:                 SARIMAX(0, 1, 1)x(1, 0, 1, 12)    Log Likelihood         370.887
Date:                  Sat, 31 Dec 2022    AIC                    -733.774
Time:                  16:26:19           BIC                    -721.923
Sample:                01-01-2003         HQIC                   -728.958
                        - 12-01-2014
Covariance Type:       opg
=====
=====
              coef    std err          z      P>|z|      [0.025    0.975]
-----
ma.L1         -0.3571     0.069     -5.191     0.000     -0.492     -0.222
ar.S.L12       0.9933     0.006    175.722     0.000      0.982      1.004
ma.S.L12      -0.5524     0.097     -5.723     0.000     -0.742     -0.363
sigma2         0.0003    2.73e-05     9.222     0.000      0.000      0.000
=====

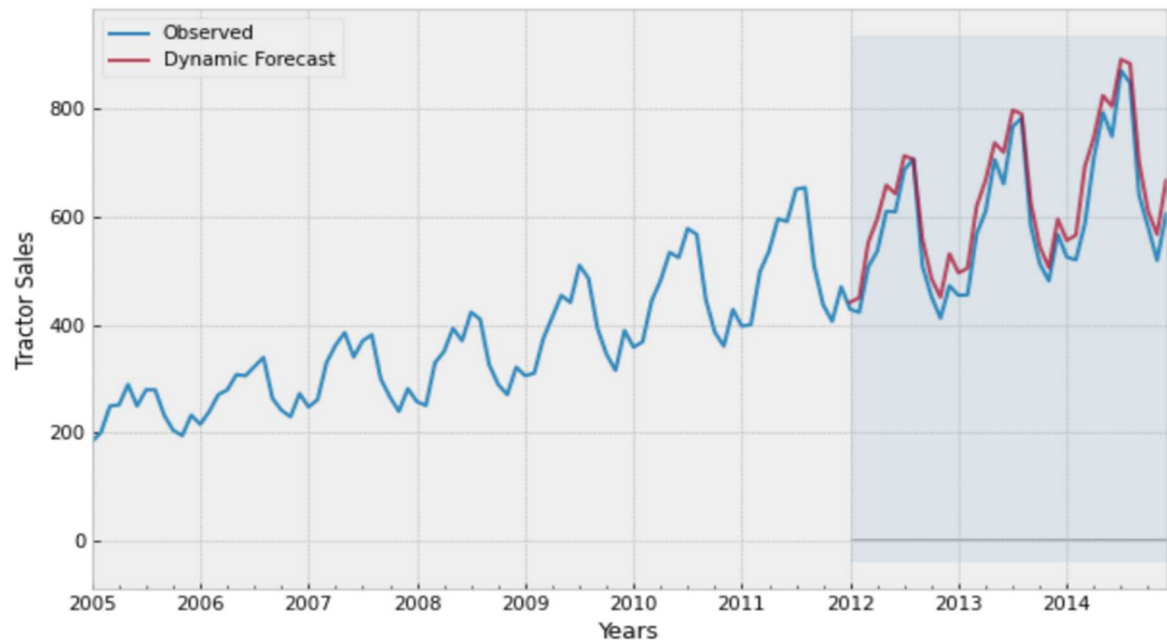
```

**Screenshot 8.9 Predict sales on sample data**

The Mean Squared Error of our forecasts is **0.0011**. By this value we choose best fit model

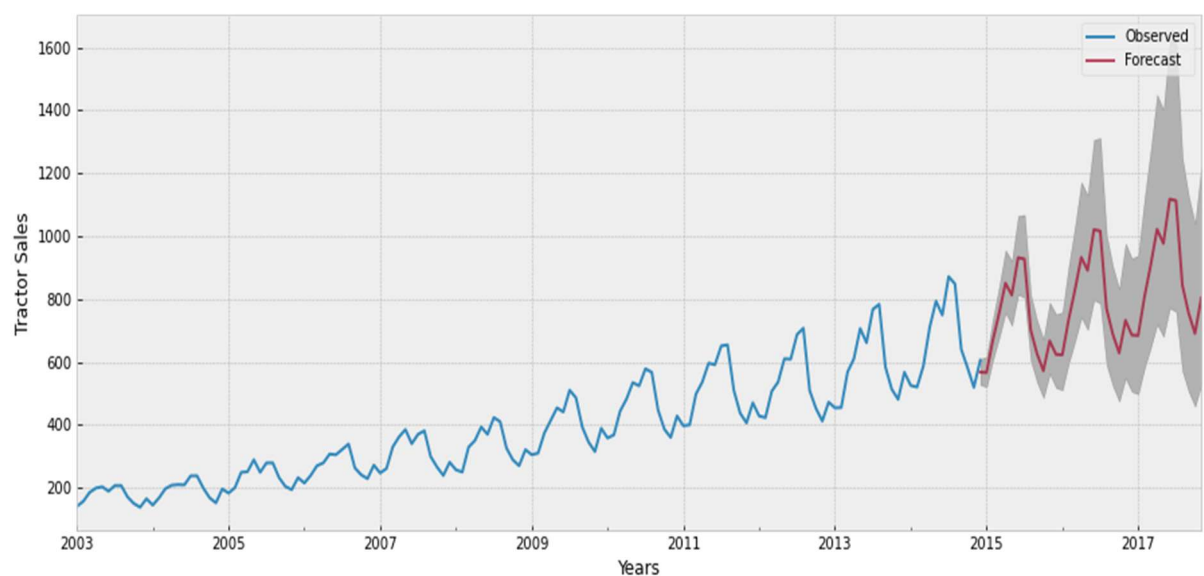
## BEST FIT ARIMA MODEL

As expected, our model has I (or integrated) component equal to 1. There is additional differencing of lag 12 in the above best fit model.



Screenshot 8.10 Best Fit ARIMA Model

## Sales Predictions (2015-2017)



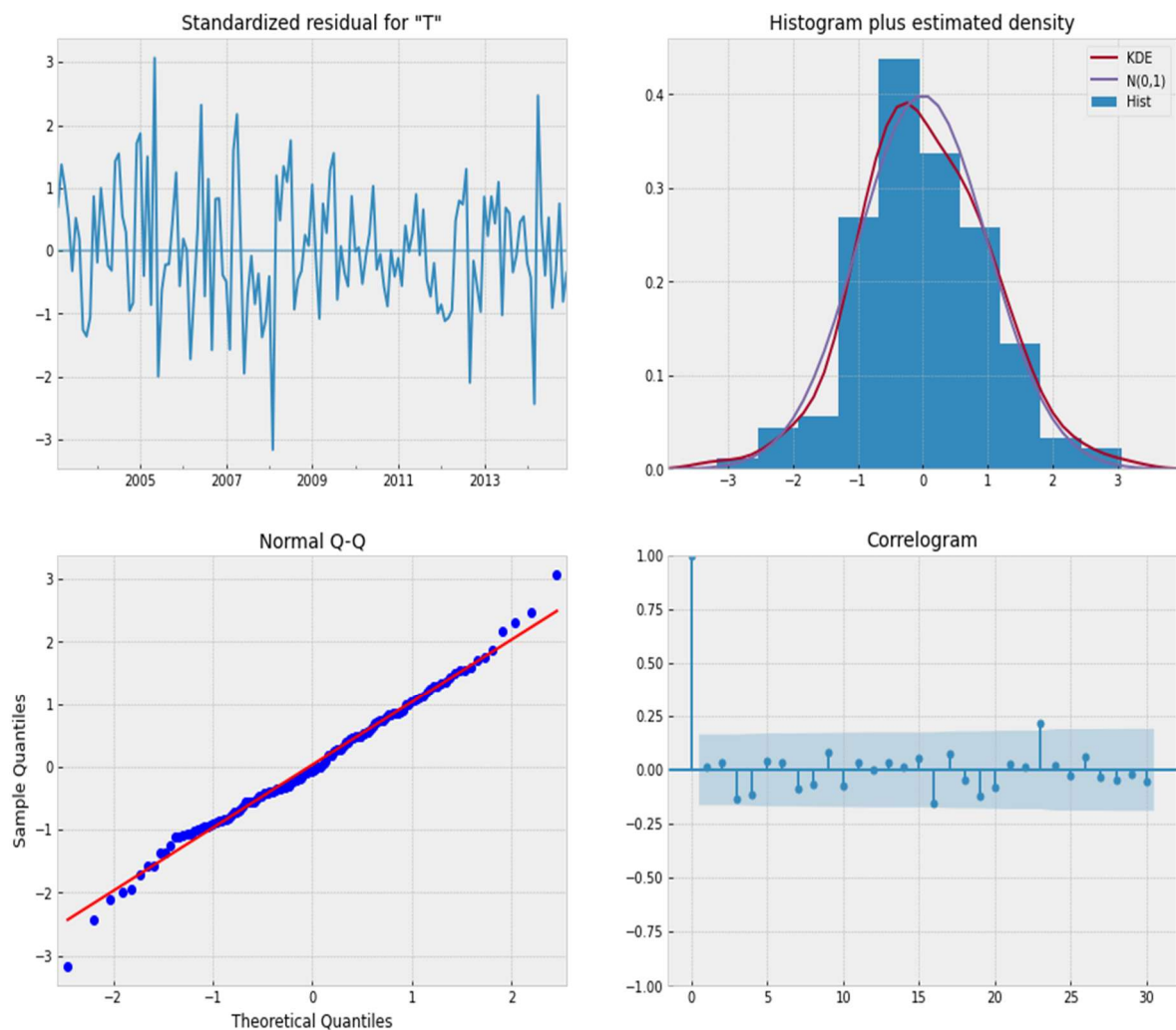
Screenshot 8.11 ARIMA Model Sales Prediction up to 2017



The final graph is plotted for the best fit ARIMA model of number of sales next following years. The following is the output with forecasted values of tractor sales in blue. Also, the range of expected error is displayed with orange lines on either side of predicted blue line.

## ACF AND PACF OF PREDICTED SALES

After predicting the number of sales, ensure that there is no more information is left for prediction that is there are no residuals in the ARIMA model.



**Screenshot 8.11 ACF and PACF of Predicted Sales**

## **9. CONCLUSION**

In this project the performance of ARIMA is done by showing different visualization graphs. Results obtained shows that ARIMA performs better for the next following years. Number of sales parameters observed that it has significant effect for ARIMA model, that is it predicts has a wide deviation from the data taken from the previous years. That is ARIMA model predicted values has a large difference from the expected values. know there no model which is predicted perfectly when it comes to sales forecasting. But as of now ARIMA model is more suitable for sales forecast for the static time series data.

## **10. FUTURE SCOPE**

Future scope of the work is to optimize the accuracy of sales forecasts and is a guide to setting a reasonable production scale. The limitations of this study are mainly reflected in the following aspects: This project only considers the effect of Weibo content text on sales prediction. As an open and interactive social platform, Weibo's comments, likes, and reposts data can further reflect the public's sentiment towards specific content. In the next step, Weibo comments, likes, and reposts will be considered in the prediction model. This project only considers the sentiment of the Weibo textual content. The emoji expressions and pictures deleted in the pre-processing stage are more likely to express the publisher's sentiment, and the next step will be to consider including the above content in the scope of sentiment analysis. The factors that affect intelligent vehicle sales are complex, such as policies, gas and electricity prices, and loan rates. In this project, only the effects of online public opinion and online search index are considered, and the next step is to fully consider various factors and further improve the prediction model. The ensemble learning method can improve the accuracy of model prediction and will be applied in future studies.

## 11. REFERENCES

- [1] Miao, X. Smart Factory and the Transformation & Upgrading of Equipment Manufacturing Industry. *Process Autom. Instrum.* 2014, 35, 1–6.
- [2] Turoń, K.; Czech, P.; Tóth, J. Safety and Security Aspects in Shared Mobility Systems. *Sci. J. Sil. Univ. Technol.* 2019, 104, 169–175.
- [3] Geva, T.; Oestreicher-Singer, G.; Efron, N.; Shimshoni, Y. Using Forum and Search Data for Sales Prediction of High-Involvement Products. *MIS Quart.* 2017, 41, 65–82.
- [4] J. A. F. D. Souza, M. M. Silva, S. G. Rodrigues, S. M. Santos, “A forecasting model based on ARIMA and artificial neural networks for end-OF-life vehicles”, *Journal of Environmental Management*, Volume 318, 2022, 115616, ISSN 0301-4797, <https://doi.org/10.1016/j.jenvman.2022.115616>.
- [5] Mehendale, Abhang, and Nadheera Sherin HR. "APPLICATION OF ARTIFICIAL INTELLIGENCE (AI) FOR EFFECTIVE AND ADAPTIVE SALES FORECASTING." *Journal of Contemporary Management Research* 12.2 (2018).
- [6] Nigam, Bhanuj, and A. C. Shukla. "Sales forecasting using Box Jenkins method based Arima model considering effect of covid-19 pandemic situation." *International Journal of Engineering Applied Sciences and Technology* (2021).
- [7] Swami, Mandeep Nim1 Kunal. "Sales Forecasting of Global Automobiles Trends using ARIMA Model."
- [8] Shakti, Sana & Hassan, Mohan & Zhenning, Yang & Caytiles, Ronnie & Iyenger, N Ch Sriman Narayana. (2017). Annual Automobile Sales Prediction Using ARIMA Model. *International Journal of Hybrid Information Technology*. 10. 13-22. 10.14257/ijhit.2017.10.6.02.
- [9] Irhami, E. Arif Farizal, F, Forecasting the Number of Vehicles in Indonesia Using Auto Regressive Integrative Moving Average (ARIMA) Method, IOP Publishing, vol. 1845, pages 1742-6596, 2021, DOI 10.1088/1742-6596/1845/1/012024.
- [10] Permatasari, Carina Intan, Wahyudi Sutopo, and Muh Hisjam. "Sales forecasting newspaper with ARIMA: A case study." *AIP Conference Proceedings*. Vol. 1931. No. 1. AIP Publishing LLC, 2018.
- [11] KAYA, Sema KAYAPINAR, and Özal YILDIRIM. "A PREDICTION MODEL FOR AUTOMOBILE SALES IN TURKEY USING DEEP NEURAL NETWORKS." *Endüstri Mühendisliği* 31.1 (2020): 57-74.

- [12] O. Haffner, E. Kučera and M. Moravčík, "Sales Prediction of Svijany Slovakia, Ltd. Using Microsoft Azure Machine Learning and ARIMA," 2020 Cybernetics & Informatics (K&I), 2020, pp. 1-9, doi: 10.1109/KI48306.2020.9039875.
- [13] Zhang, Mu, Xiaonan Huang, and Changbing Yang. "A sales forecasting model for the consumer goods with holiday effects." *Journal of Risk Analysis and Crisis Response* 10.2 (2020).