**BSE INDEX SENSEX AND NSE INDEX**

# **NIFTY-MODELLING AND FORECASTING USING TIME SERIES ANALYSIS IN R**

Project work submitted to

The Department of Statistics

# SRI KRISHNADEVARAYA UNIVERSITY

In partial fulfilment of the practical work for the degree of

MASTER OF SCIENCE IN STATISTICS , 2023



By

T.SAI VASANTH KUMAR

Project supervisor

## Dr. K. PUSHPANJALI Msc,M.phil,P.hd

HEAD OF THE

DEPARTMENT OF STATISTICS

SRI KRISHNADEVARAYA UNIVERSITY

ANANTAPURAMU- 515003

# 

# **DECLARATION**

I here by declare that the project titled “**BSE INDEX SENSEX AND NSE INDEX NIFTY-MODELLING AND FORECASTING USING R”** submitted to the Department of Statistics, SRI KRISHNADEVARAYA UNIVERSITY for the degree of Master of Science in Statistics is the result of the original work done by us. We affirm that we have not submitted this material previously for award of any degree or for any other similar purposes.

Place: Anantapuramu Signatures:

Date:

SAI VASANTH KUMAR

## 

## 

## SRI KRISHNADEVARAYA UNIVERSITY

Anantapuramu -515003



DEPARTMENT OF STATISTICS

# **CERTIFICATE**

Register No’s:

214401017

Certified that this is a benefited report of project work entitled **“BSE**

**INDEX SENSEX AND NSE INDEX NIFTY-MODELLING**

**AND FORECASTING USING R”** Was carried out by T.SAI VASANTH KUMAR

during the course of their M.Sc. (III & IV) in the year 2022-2023.

PROJECT SUPERVISOR HEAD OF THE DEPARTMENT

SIGNATURE OF THE

INTERNAL EXAMINER

## 

## ACKNOWLEDGEMENT

It gives a great pleasure to recall with a deep sense of gratitude the moments of beneficial association with our guide, friends, and other teachers and staff members. It is a great pleasure and privilege to thank **Dr.P. ADISHEKARA REDDY,** our Project

Supervisor, for helping and providing us the invaluable guidance throughout the project, we have benefied a lot from the constructive criticism and suggestions given to us by our guide. We are indebted to his constant support in the design and implementation of this project.

Also, his esteemed guidance helped us extensively

in clarifying the doubts and in execution of our computer programmes.

In addition, it is our bounded duty to thank our loving teachers, our class friends who have been helped us in lively academic discussions during the course of project activity. Also, we express our respectable regards to our parents for their cooperation in doing the project work.

We convey our sincere thanks to **Prof. K. PUSHPANJALI**

**,** Head of the Department of Statistics for her sustained support and providing all facilities in the department to complete the project work.

## 

## INDEX

## CHAPTER-I: BRIEF OVERVIEW OF STOCK MARKET IN INDIA

* 1. INTRODUCTION

1.2 BOMBAY STOCK EXCHANGE (BSE)

1.3 NATIONAL STOCK EXCHANGE (NSE)

1.4 LITERATURE REVIEW

1.5 MOTIVATION

**CHAPTER-II: TIME SERIES ANALYSIS – METHODOLOGY**

2.1 INTRODUCTION TO TIME SERIES

2.2 HOLT'S DOUBLE EXPONENTIAL SMOOTHING

METHOD FOR ANALYSIS AND FORECASTING OF TIME SERIES

2.3 AUTOREGRESSIVE MOVING AVERAGE MODEL (ARMA)

2.4 AUTOREGRESSIVE INTEGRATED MOVING

AVERAGE (ARIMA) MODEL

**CHAPTER-III: BSE INDEX SENSEX -**

**MODELING AND FORECASTING USING TIME SERIES ANALYSIS**

**3.1 INTRODUCTION TO R-PROGRAMMING LANGUAGE**

**3.2 DATA COLLECTION**

**3.3 DATA ANALYSIS**

3.2.1Analysis of monthly SENSEX indices of BSE

3.2.2Analysis of weekly SENSEX indices of BSE

**CHAPTER-IV: NSE INDEX NIFTY-**

**MODELING AND FORECASTING USING TIME SERIES ANALYSIS**

**4.1 DATA COLLECTION**

**4.2 DATA ANALYSIS**

4.2.3Analysis of monthly NIFTY indices of NSE

4.2.4Analysis of weekly NIFTY indices of NSE

**CHAPTER-V:** **OBSERVATION AND CONCLUSIONS**

5.1 CONCLUSION ON BSE

5.2 CONCLUSIONS ON NSE

5.3 FURTHER SCOPE

5.4 REFERENCES

**CHAPTER – I**

**BOMBAY STOCK EXCHANGE AND NATIONAL STOCK EXCHANGE – PROFILES OF THE COMPANIES AIM OF THE PROJECT:**

The aim of this project is to analyze the monthly and weekly indices (BSE SENSEX and NSE NIFTY) of the two reputed Indian Stock Market Exchanges namely Bombay Stock Exchange (BSE) and National Stock

Exchange (NSE) using some suitable time series analysis techniques. Generally, in the stock market indices there will be some sudden ups and downs, which are due to some unexpected important factors like government policy matters, the performance of different multinational companies including banking sectors, central budget, and etc.. Though there are some seasonal influences, they are negligible. Therefore, we chose only nonseasonal time series analyses models for the analysis of the indices of BSE-

SENSEX and NSE-NIFTY and we explain two such models viz.,

1. Moving average
2. HOLT’S SINGLE and DOUBLE exponential model &
3. ARIMA model

in CHAPTER-II of this project dissertation.

**Using the above these time series analysis methods, we analyze the past data of the indices of the two reputed Indian stock market exchanges BSESENSEX and NSE-NIFTY and we forecast the indices of both stock exchanges for the future years, using the better model**.

**1.1 INTRODUCTION TO STOCK MARKET IN INDIA**

The stock market in India plays a crucial role in the country's economy by providing a platform for companies to raise capital and for investors to buy and sell ownership stakes in those companies. It is a key component of the financial system and offers individuals and institutions opportunities to participate in the growth and profitability of Indian businesses. Here's a brief introduction to the stock market in India:

**1**. **Regulatory Bodies**: The primary regulatory bodies governing the Indian stock market are the Securities and Exchange Board of India (SEBI) and the Reserve Bank of India (RBI). SEBI is responsible for overseeing the functioning of stock exchanges, protecting the interests of investors, and ensuring fair and transparent operations.

**2. Stock Exchanges**: The two major stock exchanges in India are the Bombay Stock Exchange (BSE) and the National Stock Exchange (NSE). These exchanges facilitate the trading of various financial instruments, including equities (stocks), derivatives, bonds, and other securities.

**3. Equity Shares**: Equity shares, also known as stocks or shares, represent ownership in a company. When you buy shares of a company, you become a shareholder and hold a portion of that company. Shareholders may benefit from dividends (a share of company profits) and potential capital appreciation if the company's stock price increases.

**4. Initial Public Offering (IPO):** When a company decides to go public, it offers its shares to the general public for the first time through an IPO. Investors can buy these shares, thus becoming shareholders of the company. IPOs provide companies with an opportunity to raise capital for expansion and growth.

**5. Secondary Market**: The secondary market is where existing shares are traded among investors. This is where most stock market activity takes place. Investors can buy and sell shares of listed companies on the stock exchanges.

**6. Indices**: Stock market indices are benchmarks that track the performance of a group of stocks. In India, the most well-known indices include the BSE Sensex (BSE 30) and the Nifty 50 (NSE 50), which represent the overall performance of the top companies listed on BSE and NSE, respectively.

**7. Trading Mechanism**: Stock trading in India occurs through a computerized system known as electronic trading platforms. Orders are matched electronically based on prices and trading volumes.

**8. Market Participants**: The stock market involves various participants, including individual retail investors, institutional investors (mutual funds, pension funds, etc.), foreign institutional investors (FIIs), market makers, and stockbrokers.

**9. Risk and Return**: Investing in the stock market offers the potential for high returns, but it also involves risks. Stock prices can be volatile, and there's no guarantee of profits. It's important for investors to conduct thorough research, diversify their portfolio, and have a long-term perspective.

**10. Dematerialization**: Physical share certificates have been largely replaced by dematerialized (demat) accounts. Investors hold their securities in electronic form, which enhances convenience, reduces paperwork, and simplifies trading.

**11. Investor Protection**: SEBI has put in place various regulations and measures to protect the interests of investors, ensure fair trading practices, and maintain market integrity.

**12. Online Trading**: The advent of the internet has made it possible for investors to trade stocks online through brokerage platforms, providing easy access to the stock market.

Before investing in the Indian stock market, it's advisable to educate yourself about different investment options, understand the market dynamics, and consider seeking advice from financial professionals. It's also important to have a clear investment strategy and risk tolerance in mind.

### Bombay Stock Exchange (BSE)

The Bombay Stock Exchange (BSE) is the first and largest securities market in India and was established in 1875 as the Native Share and Stock Brokers' Association. Based in Mumbai, India, the BSE lists close to 6,000 companies and is one of the largest exchanges in the world, along with the New York Stock Exchange (NYSE), Nasdaq, London Stock Exchange Group, Japan Exchange Group, and Shanghai Stock Exchange.

The BSE has helped develop India's capital market, including the retail debt market, and has helped grow the Indian corporate sector. The BSE is Asia's first stock exchange and also includes an equities trading platform for small-andmedium enterprises (SME). BSE has diversified into providing other capital market services including clearing, settlement, and risk management.

**History**

Bombay Stock Exchange was started by Premchand Roychand in 1875. While BSE Limited is now synonymous with Dalal Street, it was not always so.

In the 1850s, five stock brokers gathered together under a Banyan tree in front of Mumbai Town Hall, where Horniman Circle is now situated. A decade later, the brokers moved their location to under the banyan trees at the junction of Meadows Street and what was then called Esplanade Road, now Mahatma Gandhi Road. With a rapid increase in the number of brokers, they had to shift places repeatedly. At last, in 1874, the brokers found a permanent location, the one that they could call their own. The brokers group became an official organization known as "The Native Share & Stock Brokers Association" in 1875.

On 12 March 1993, a car bomb exploded in the basement of the building during the 1993 Bombay bombings. The BSE is also a Partner Exchange of the

United Nations Sustainable Stock Exchange initiative, joining in September 2012. BSE established India INX on 30 December 2016. India INX is the first international exchange of India. BSE became the first stock exchange in the country to launch commodity derivatives contract in gold and silver in October 2018.

BSE was demutualized and corporatized on 19 May 2007, pursuant to the BSE (Corporatization and Demutualization) Scheme, 2005 notified by SEBI. It was listed on NSE on 3 February 2017

### KEY TAKEAWAYS

* Established in 1875 as the Native Share and Stock Brokers' Association, the Bombay Stock Exchange (BSE) is Asia's first exchange and the largest securities market in India.
* The BSE has been instrumental in developing India's capital markets by providing an efficient platform for the Indian corporate sector to raise investment capital.
* The BSE is known for its electronic trading system that provides fast and efficient trade execution.
* The BSE enables investors to trade in equities, currencies, debt instruments, derivatives, and mutual funds.
* The BSE also provides other important capital market trading services such as risk management, clearing, settlement, and investor education.

### How the Bombay Stock Exchange (BSE) Works

In 1995, the BSE switched from an open-floor to an electronic trading system. There are more than a dozen electronic exchanges in the U.S. alone with the New York Stock Exchange (NYSE) and Nasdaq being the most widely known.

Today, electronic trading systems dominate the financial industry overall, offering fewer errors, faster execution, and better efficiency than traditional open out-cry trading systems. Securities that the BSE lists include stocks, stock futures, stock options, index futures, index options, and weekly options.

The BSE's overall performance is measured by the Sensex, a benchmark index of 30 of the BSE's largest and most actively traded stocks covering 12 sectors. Debuting in 1986, the Sensex is India's oldest stock index. Also called the "BSE 30," the index broadly represents the composition of India's entire market.

### Functions of BSE

The following are the primary functions of the Bombay Stock Exchange –

**Price determination**: The prices of securities in the secondary market depend on the securities’ demand and supply. Thus, BSE helps in this process by constantly valuing all the listed securities. And investors can easily track the prices of these securities through the index popularly known as SENSEX.

**Contribution the economy:** BSE offers a for trading perform securities of various companies. The trading process involves continuous reinvestment and disinvestment. This gives an opportunity for capital formation, funds movement and boosting of the economy.

**Facilities liquidity:** The most important function of BSE is ensuring a ready platform for the sale and purchase of securities. This gives investors the confidence to convert the existing securities into cash anytime. Thus, investors can buy and sell anytime offering them high liquidity.

**Transactional safety:** BSE ensures that the securities are listed after verifying the company’s position. Also, all listed companies must adhere to the rules and regulations laid out by the governing body, i.e. securities and control board of India.

**Advantages of listing with BSE:**

There are several advantages that a company can enjoy by listing with BSE –

**Easy generation of capital:** The companies listed on the stock exchange build confidence among all investors. Also, it spreads market knowledge about business, allowing investors to properly assess the company’s future prospects and invest accordingly. A company can raise the paid-up capital effectively only if it is there on the country’s major stock exchange. Additionally, securities listed on the BSE can easily be bought and sold in the financial market, thereby providing sufficient liquidity to both businesses and investors. The companies can raise funds for any business requirement by issuing equity or debt security which investors purchase for the purpose of wealth generation. Investors can also sell their securities using the BSE’s electronic trading settlement, allowing investors to encash their investment as and when the need arises.

**Legal Supervision:** If the investors choose to invest in companies listed on BSE, they can skim through the fraudulent companies. Also, SEBI governs several rules and regulations to monitor the operations of these listed companies. Thus, it minimizes the chances of investors losing money due to the company’s illegal activity.

**Publishing Adequate Information**: The companies on the Bombay Stock Exchange must report adequate information about total revenue generation and reinvestment patterns on an annual basis. As per SEBI regulations, companies must also report total dividends, bonus and transfer issues, and the book-toclosure facility.

**Pricing Rules**: The current supply and demand determine the pricing of securities trading in the BSE stock market. This price reflects the real value of the share, which affects the company’s market capitalisation and ease of procurement of funds. While availing of loans, the company’s security serves as a collateral guarantee. Also, most financial institutions accept equity shares listed on BSE as collateral for obtaining funds.

**Investment and Trading Segments**: The securities listed on the Bombay Stock Exchange can be traded directly or indirectly, depending on the volume of transactions. Primarily bulk transactions on BSE are only possible through registered brokerage agencies and institutional investors. On the other hand, retail investors can make transactions through certified stock brokers, broking firms, or any stock investing platform. Thus, Financial Industry Regulatory Authority (FINRA) regulates secondary trading in India. Investors must open a demat account for the financial transactions under this mechanism.

**Major Indices in BSE:**

The primary index of the Bombay Stock Exchange is Sensex. It is a free float market-weighted index that tracks the performance of the top 30 companies in India. Therefore, the BSE share market uses Sensex to track the performance of these companies to determine whether the Indian capital market will fall or rise depending on the stock price movement.

Apart from the benchmark index, BSE offers other sectoral indices. Some of them are –

S&P BSE Auto

S&P BSE Bankex

S&P BSE Capital Goods

S&P BSE Consumer Durables

S&P BSE Fast Moving Consumer Goods

Additionally, BSE has established indices that divide companies into small and mid-cap based on their Market capitalization. They are known as the BSE small-cap index and BSE midcap index. Moreover, index mutual funds that seek to profit from the capital appreciation of these companies can monitor these indices.

**SENSEX**

The Sensex is one of the oldest stock exchanges of India. It comprises total value of 30 stocks of companies which are listed on the BSE. Indeed, these stocks belong to the largest corporations in India and, thus, represent the Indian economy’s performance at large.

To simplify, if the Sensex is moving upwards, then the investors or traders in the market will prefer buying stocks and on the other hand, if the Sensex is moving downward, the investors or trader will prefer to hold back their positions. The Sensex movements are tracked regularly which helps in analyzing the overall growth, industry-related development.

Below are the criteria which are used in selecting the 30 stocks of the Sensex:

* Stock must be listed on the BSE.
* Large-cap stocks with high market capitalization.
* High liquidity.
* Average daily turnover.
* Wide industry representation.

**National Stock Exchange (NSE):**

**National Stock Exchange** (**NSE**) is one of the leading stock exchanges in India, based in Mumbai. NSE is under the ownership of various financial institutions such as banks and insurance companies. It is the world's largest [derivatives exchange](https://en.wikipedia.org/wiki/Derivatives_exchange) by number of contracts traded and the third largest in cash equities by number of trades for the calendar year 2022. It is one of the [largest stock exchanges](https://en.wikipedia.org/wiki/List_of_stock_exchanges) in the world by market capitalization. NSE's flagship index, the [NIFTY 50,](https://en.wikipedia.org/wiki/NIFTY_50) a 50-stock index is used extensively by investors in India and around the world as a barometer of the Indian capital market. The **NIFTY 50** index was launched in 1996 by NSE.

[The Economic Times](https://en.wikipedia.org/wiki/The_Economic_Times) estimates that as of April 2018, 6 crore (60 million) retail investors had invested their savings in stocks in India, either through direct purchases of equities or through mutual funds Earlier, the [Bimal Jalan](https://en.wikipedia.org/wiki/Bimal_Jalan) Committee report estimated that barely 3% of India's population invested in the stock market, as compared to 27% in the [United States](https://en.wikipedia.org/wiki/United_States) and 10% in [China.](https://en.wikipedia.org/wiki/China)

**History**

National Stock Exchange was incorporated in the year 1992 to bring about transparency in the Indian equity markets. NSE was set up at the behest of the [Government of India,](https://en.wikipedia.org/wiki/Government_of_India) based on the recommendations laid out by the [Pherwani committee](https://en.wikipedia.org/wiki/Manohar_J._Pherwani) in 1991and the blueprint was prepared by a team of five members (Ravi Narain, Raghavan Puthran, K Kumar, Chitra Sankaran and [Ashishkumar Chauhan)](https://en.wikipedia.org/wiki/Ashishkumar_Chauhan) along with Dr. R H Patil and [SS Nadkarni](https://en.wikipedia.org/wiki/Suresh_Shankar_Nadkarni) who were deputed by IDBI in 1992. Instead of trading memberships being confined to a group of brokers, NSE ensured that anyone who was qualified, experienced, and met the minimum financial requirements was allowed to trade.

NSE commenced operations in 30 June 1994 starting with the wholesale debt market (WDM) segment and equities segment in 03 November 1994. It was the first exchange in India to introduce an [electronic trading facility.](https://en.wikipedia.org/wiki/Electronic_trading_platform) Within one year of the start of its operations, the daily turnover on NSE exceeded that of the [BSE.](https://en.wikipedia.org/wiki/Bombay_Stock_Exchange)

Operations in the derivatives segment commenced in 12 June 2000. In August 2008, NSE introduced [currency derivatives.](https://en.wikipedia.org/wiki/Foreign_exchange_derivative)

**KEY TAKEAWAYS**

* The National Stock Exchange of India Limited (NSE) is India's largest financial market and the fourth largest market by trading volume.
* The National Stock Exchange of India Limited was the first exchange in India to provide modern, fully automated electronic trading.
* The NSE is the largest private wide-area network in India.
* The NSE has been a pioneer in Indian financial markets, being the first electronic limit order book to trade derivatives and ETFs.

**Understanding the National Stock Exchange (NSE)**

Today, the National Stock Exchange of India Limited (NSE) conducts transactions in the wholesale debt, equity, and derivative markets. One of the more popular offerings is the NIFTY 50 Index, which tracks the largest assets in the Indian equity market. US investors can access the index with [exchangetraded funds](https://www.investopedia.com/terms/e/etf.asp) (ETF), such as the iShares India 50 ETF (INDY).

The National Stock Exchange of India Limited was the first exchange in India to provide modern, fully automated electronic trading. It was set up by a group of Indian financial institutions with the goal of bringing greater transparency to the Indian capital market.

### Benefits of the NSE

The National Stock Exchange is a premier marketplace for companies preparing to list on a major exchange. The sheer volume of trading activity and application of automated systems promotes greater transparency in trade matching and the settlement process.

This in itself can boost visibility in the market and lift investor confidence. Using cutting-edge technology also allows orders to be filled more efficiently, resulting in greater liquidity and accurate prices.

**Functions of NSE**

The NSE was set-up with an express objective to fulfil the following functions:

establishing a nation-wide trading facility for equities, debt and other hybrid instruments ensuring equal access to investors across the nation through an appropriate communication network providing a fair, efficient and transparent securities market to investors using electronic trading systems enabling [shorter settlement cycles and book entry settlements systems,](https://www.samco.in/knowledge-center/articles/what-are-unsettled-holdings-or-t1-holdings/) and meeting the current international standards of securities markets

NSE successfully fulfilled these functions by establishing the first electronic stock market of the nation. NSE was instrumental in creating National Securities Depository Limited (NSDL), [the first depository in India,](https://www.samco.in/knowledge-center/articles/what-is-a-depository/) allowing investors to hold and trade securities electronically. This not only made investing simple, but also provided increased transparency. The price information that was earlier available only to a handful of traders present at the exchange, was now widely broadcasted and available to everyone at their own remote location.

Before the system introduced by NSE, an investor who wanted to trade a security not listed on the nearest exchange had to route orders through a series of correspondent brokers to the appropriate exchange. This resulted in increased uncertainty and high transaction costs. NSE made it possible for an investor to access the same market and order book, irrespective of location and at the same cost as every other investor. NSE trading terminals are now present in 363 cities and towns across India and can be accessed through brokers from anywhere on the globe.

**Features of National Stock Exchange**

NSE, like every other leading stock exchange today, runs an order-driven market as opposed to quote-driven market. The fully automated screen-based trading system that it runs is called National Exchange for Automated Trading (NEAT).

The order management system under NEAT gives a unique number to each order received and if a match is not found immediately, it is added to an order book where the sequence of orders to be matched are determined based on price-time priority. That is, if two orders are entered into the system, the order having the best price gets the higher priority and within the orders of the same price priority is given to the older order.

Order matching is done by comparing the best buy order, the buy order with the highest price, with the best sell order, the sell order with the lowest price. This is because a seller would like to sell to the buyer offering the highest price and vice versa. While orders can be partially matched till the complete order can be completed, the matches are always made based on the passive price of the order and not the active price at which the match is made.

NEAT also allows members to specify conditional clauses on the submitted orders These clauses can be of the following kinds:

* Time related condition Price related condition
* Quantity related condition

### NIFTY 50

The NIFTY 50 is the flagship index of the National Stock Exchange and one of the most recognized stock market indexes of India. It tracks the total of 50 stocks of huge companies related to various sectors and industries. The NIFTY 50 based stocks are all large-cap oriented companies which form almost three-fourth of the total capitalization in India. NIFTY 50 helps in benchmarking fund portfolios, launching of index funds, ETFs and other structured products.

Below are the criteria which are used in selecting the 50 stocks of NIFTY 50:

* Stock must be listed on the NSE and should be included in NSE’s futures and options trading list.
* Company’s registered office should be in India.
* Large-cap stocks with market capitalization up to INR
* High liquidity
* High volume

**Difference Between Sensex and NIFTY 50?**

The Nifty 50 and Sensex sound a lot similar to each other and thus are often used for the same purpose. But in practicality, they are quite different from each other. Let’s delve more into these two indices- Sensex and **Nifty50** to know the differences between them.

**LITERATURE REVIEW :**

A brief literature review on the topic of time series forecasting for BSE (Bombay Stock Exchange) and NSE (National Stock Exchange) using the R language.

**1. Pandya, D., & Bhatt, N.(2016)**. "A Comparative Study of Time Series Forecasting Models for Stock Market Prediction":

This comprehensive study aimed to compare the performance of several time series forecasting models applied to stock market data from the Bombay Stock Exchange (BSE) and National Stock Exchange (NSE). The authors explored traditional models such as AutoRegressive Integrated Moving Average (ARIMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH), along with more advanced techniques like Neural Networks. They utilized the R language for implementing these models and evaluated their accuracy using various metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). The study's findings provided insights into the strengths and weaknesses of different models in capturing the complex dynamics of stock market data.

**2. Mishra, A., & Sahoo, J. (2018).** "Time Series Forecasting Using ARIMA Model on NSE Nifty":

Focusing specifically on the NSE Nifty stock index, this study delved into the application of the ARIMA model for time series forecasting. The researchers acquired historical NSE Nifty data and used the R language to preprocess the data, estimate the ARIMA model parameters, and generate forecasts. The study discussed the Box-Jenkins methodology for model selection and emphasized the iterative process of refining the model orders to achieve better predictions. The findings highlighted the importance of understanding the inherent characteristics of the data and selecting appropriate model parameters for accurate forecasting.

**3. Raval, H. K., & Shah, R. R. (2019).** "Predictive Analysis of Stock Market Data using Time Series Models":

This study contributed valuable insights into the predictive analysis of BSE and NSE stock prices using time series models. Employing the R language, the authors demonstrated the application of ARIMA and GARCH models to forecast stock prices and volatility. They discussed the challenges posed by the stock market's inherent volatility and its sensitivity to external factors such as economic indicators and geopolitical events. The research highlighted the need for robust modeling techniques that can adapt to changing market conditions.

**4. Sharma, A., & Sharma, S.(2020).** "Forecasting of NSE Nifty using ARIMA Model":

Focused on the NSE Nifty index, this study provided a detailed exploration of the ARIMA model for time series forecasting. The researchers utilized the R language to implement the ARIMA model, conduct diagnostic checks, and evaluate forecast accuracy. The study underscored the importance of diagnostics to ensure the adequacy of the model assumptions and discussed the challenges of handling sudden market movements and shifts in trend. The findings emphasized the necessity of continuous model refinement to enhance predictive accuracy.

**5. Jain, P., & Kumar, A.(2021).** "A Comparative Study of Machine Learning Algorithms for Stock Market Prediction":

In response to the growing interest in machine learning techniques, this study conducted an in-depth comparison of various algorithms, including traditional time series methods like ARIMA and more modern techniques like Random Forest, Support Vector Machines (SVM), and XGBoost. The study applied these algorithms to predict BSE and NSE prices, using R for implementation and evaluation. The findings highlighted the distinct advantages of each method and emphasized their complementary nature in capturing different facets of stock market behavior.

**6. Bharati, M., & Chakraverty, S. (2021).** "Forecasting Stock Market Using Time Series Models: A Case Study on NSE Nifty 50 Index":

Focusing exclusively on the NSE Nifty 50 index, this research explored the application of time series models such as ARIMA and Exponential Smoothing for forecasting. The study employed the R language to fit the models to historical data and generate forecasts. It discussed the significance of model selection based on forecast accuracy and highlighted the challenges of predicting stock market indices due to their sensitivity to macroeconomic factors and external shocks.

The aforementioned studies collectively contribute to the body of knowledge related to time series forecasting for BSE and NSE using the R language. They delve into various aspects of data preprocessing, model selection, parameter tuning, and performance evaluation. While these studies offer valuable insights, it's important to recognize that stock market forecasting remains a complex and inherently uncertain task, influenced by a multitude of economic, political, and behavioural factors.

**MOTIVATION :**

Motivation for Modeling and Forecasting BSE Sensex and NSE Nifty Using Time Series Analysis in R:

**1. Financial Decision-Making Advancement:**

The BSE Sensex and NSE Nifty are key indicators of the Indian stock market's overall performance. Accurate forecasts of these indices empower investors, fund managers, and traders to make well-informed decisions about portfolio allocation, investment strategies, and risk management. Effective models provide insights into potential market movements, enabling stakeholders to maximize returns and minimize losses.

**2. Risk Management Enhancement:**

The financial markets are subject to fluctuations driven by various factors like economic events and geopolitical developments. Developing reliable time series forecasting models helps individuals and institutions anticipate and manage risks associated with market volatility. Timely identification of potential downturns can lead to proactive risk mitigation strategies.

**3. Macroeconomic Insights:**

The movements of BSE Sensex and NSE Nifty reflect broader economic trends. Accurate forecasts can help policymakers, economists, and analysts gain insights into economic health and potential future directions. Time series analysis provides a means to uncover relationships between market indices and economic indicators, contributing to a deeper understanding of economic dynamics.

**4. Investor Confidence Boost:**

Reliable forecasts inspire investor confidence. When stakeholders see accurate predictions that align with actual market movements, their trust in financial analysis and predictions grows. Enhanced trust encourages more informed investment decisions, stabilizing the market and promoting healthy investment activity.

**5. Portfolio Diversification and Allocation:**

Effective time series models offer insights into the performance of various sectors and industries. Investors can use these forecasts to diversify portfolios and allocate resources based on predicted sectoral performance. This optimization leads to a well-balanced portfolio and minimizes overexposure to underperforming sectors.

**6. Algorithmic Trading Strategies:**

In today's digital age, algorithmic trading strategies are prevalent. Accurate time series forecasts of BSE Sensex and NSE Nifty enable algorithmic traders to develop strategies that capitalize on predicted price movements. These strategies contribute to market liquidity and efficiency.

**7. Academic and Professional Research:**

Research in time series modeling and forecasting contributes to both academia and professional practice. Academically, it advances the field of financial econometrics and time series analysis, leading to new methodologies and insights. Practically, the outcomes of research benefit analysts, financial professionals, and institutions in making data-driven decisions.

**8. Technological Advancements:**

The availability of historical market data and powerful computational tools like R allows researchers to develop and implement sophisticated models. Researchers can employ advanced methodologies such as ARIMA, GARCH, or even machine learning algorithms to capture intricate market dynamics and enhance predictive accuracy.

**9. Competitive Edge for Traders and Funds:**

In the competitive world of trading and investment management, accurate forecasting models offer a strategic advantage. Hedge funds, asset managers, and traders who possess reliable models can achieve superior returns and distinguish themselves from peers.

**10. Personal Financial Planning:**

On an individual level, accurate forecasts of BSE Sensex and NSE Nifty assist retail investors in making decisions related to retirement planning, goal-setting, and wealth accumulation. Reliable predictions enable individuals to adjust their investment strategies based on market expectations.

Modeling and forecasting BSE Sensex and NSE Nifty using time series analysis in R serve the broader goals of informed decision-making, risk management, economic insights, investor confidence, and academic advancement. These motivations collectively contribute to the betterment of financial markets, economic understanding, and individual financial well-being.

**CHAPTER-II**

**TIME SERIES ANALYSIS – METHODOLOGY**

**1. INTRODUCTION:**

A Time series is a sequence of observations taken sequentially in time. Many sets of data appear as Time series: a monthly sequence of a quantity of goods shipped from a factory, a weekly series of the number of road accidents, hourly observations made on in the field of a chemical process, and so on. An intrinsic feature of a Time series Is that typically adjacent observations are dependent. The nature of this dependence among observations of time series is of considerable practical interest. Unlike the analyses of random samples of observations that is discussed in the context of most other statistics, the analysis of time series is based on the assumption that successive values in the data file represent consecutive measurements taken at equally spaced time intervals. Time series analysis is concerned with techniques for the analysis of this dependence. This requires the development of stochastic and dynamics data and the use of such models in important areas of models for time series applications. There are two kinds of time series data:

1. Continuous, where we have an observation at every instant of time, e.g. lie detectors, electrocardiograms. We denote this using observation Y at time t, Y(t).
2. Discrete, where we have an observation at (usually regularly) spaced Intervals. We denote this as Yt.

Examples:

Economics - weekly share prices, monthly profits

Meteorology - daily rainfall, wind speed, temperature

Sociology - crime figures (number of arrests, etc)

There are two main goals of time series analysis: (a) identifying the nature of the phenomenon represented by the sequence of observations, and (b) forecasting (predicting future values of the time series variable). Both of these goals require that the pattern of observed time series data is identified and more or less formally described. Once the pattern is established, we can interpret and integrate it with other data (i.e., use it in our theory of the investigated phenomenon, €. g., seasonal commodity prices). Regardless of the depth of our understanding and the validity of our interpretation (theory) of the phenomenon, we can extrapolate the identified pattern to predict future events.

**Components of Time Series:**

In general, there are four types of components in time series analysis: Trend, Seasonality, Cycling and Irregularity. An examination of what these components are, how they are, formulated, how they may differ from one to another, how one tests for their presence, and how one estimates their parameters is in order. When early economists sought to understand the nature of a business cycle, they began studying series In search of calendar effects,

‘trends, cycles, seasonal and irregular components. These components could be added or multiplied together to constitute the time series. The decomposition could be represented by

**Additive Model:** Ŷat = T̂ t + Ŝt + Ĉt + Ît

**OR**

**Multiplicative Model:** Ŷmt = T̂ t × Ŝt × Ĉt × Ît where Ŷat = additively composed time series,

Ŷmt = multiplicatively composed time series,

T̂ t = trend,

Ŝt = seasonality,

Ĉt = Cyclicity,

Ît = irregularity.

Whether the process under guiding an observed series is additive or multiplicative, one needs to ascertain whether it contains a trend.

**Trend Component:**

We want to increase our understanding of a time series by picking out its main features. One of these main features is the trend component. Descriptive techniques may be extended to forecast (predict) future values. A time series may be stationary or exhibit trend over time. Long-term trend is typically modelled as a linear, quadratic or exponential function. Trend is a long-term movement in a time series. It is the underlying direction (an upward or downward tendency) and rate of change in a time series, when allowance has been made for the other components. A simple way of detecting trend in seasonal data Is to take averages over a certain period, if these averages change with time, we can say that there is evidence of a trend in the series There are also more formal tests to enable detection of trend in time series.

Trends, whether deterministic or stochastic, have to be considered for extracting, fitting and forecasting, A deterministic trend may derive from a definition that prescribes a well-defined formula for increment or decrement as a function of time, such as contractual. The cost of a three-year loan may increase by agreement at a simple 2% per year. The interest on the loan by agreement is 0.02% per year. The amount of interest in effect is determined by agreement on formula and hence deterministic.

A stochastic trend is due to random shift of level, perhaps the cumulative effect of some force that endows the series with a long-run change in level. Trends may stem from changes in society, social movements, technology, social custom, economic conditions, market conditions, or environment. As the trend represents a shift of the mean, it needs to be detected, identified and modelled or the series may become unamenable to modelling, fitting and forecasting. Usually, the series can be detrended by decomposing it in to its components of variations and extracting these signals.

Regression may be used to test and model a trend. First, one plots the series against time. If the trend appears linear, one can regress it against a measure of time. If one finds a significant or substantial relationship with time, the magnitude of the coefficient of time is evidence of linear trends. Alternatively, some trends may appear to be a non-linear. When non-linear relationships exist, one can transform them in to linear ones prior to modelling by either a log transformation of the dependent variable or a Box-Cox transformation. The series may be trended by regression or transformation.

If the functional form of the trend is more complicated, the researcher designates the real data as the series of Interest C(t) and functional form Y(t). we may compute the sum of squared errors (SSE)as follows: SSE=[C(t)-Y(t)]2. This is the unexplained sum of squares. The proportion of variance explained for the model, R2, may be computed as follows R2 = 1- (SSE/SS Total). R2 is the objective criterion by which the fit is tested the R2 and significance test for each parameter indicate which are significant. The functional form with the highest R2 indicates the best fit and the one that should be chosen. Many monotonous time series data can be adequately approximated by a linear function; if there is a clear monotonous non-linear component, the data first need to be transformed to remove the nonlinearity. Usually a logarithmic, exponential, or (less often) polynomial function can be used. There are no proven "automatic" techniques to identify trend components in the time series data; however, as long as the trend is monotonous (consistently Increasing or decreasing) that part of data analysis is typically not very difficult. If the time series data contain considerable error, then the first step in the process of trend identification is smoothing.

**Seasonal Component:**

When a repetitive pattern is observed over some time horizon, the seriesis said to have seasonal behaviour. Seasonal effects are usually associated withcalendar or climatic changes. Seasonal variation is frequently tied to yearlycycles. When the series is characterized by a substantial regular annualvariation, one must control for the seasonality as well as trend in order toforecast. Seasonality, the periodic annual changes in the series, may follow fromyearly changes in whether such as temperature, humidity, or precipitation. Itdescribes any regular fluctuations with a period of less than one-year Seasonalchanges provide optimal times in the crop cycle for turning the soil, fertilizing,planting, and harvesting. Summer vacations from primary and secondary schooltraditionally allow children time for summer recreation. Sports equipment andclothing sales in temperate zones follow the seasons, whether for water or Snowsports. Forecasting with such series requires seasonal adjustments or seasonalvariation may augment the forecast error unnecessarily.

We are interested in comparing the seasonal effects within the years, from year to year; removing seasonal effects so that the time series is easier to cope with; and, also interested in adjusting a series for seasonal effects using various models.

**Cyclical Component:**

An upturn or downturn not tied to seasonal variation. Usually results from changes in economic conditions. For years economists have searched for clear cut-cycles, like those found in nature-for example the sun spot cycle. Economists have searched for inventory, investment, growth, building and monetary cycles. Eventually, researchers began to look for indicators of the business cycle. They searched for leading indicators that would portend a turning point in the business cycle. Although they found a number of coincident and lagging indicators, the search for reliable leading indicators has generally been unsuccessful. Where trend unicycle or not separated from one another, the series components is called the trend-cycle. Descriptive techniques may be extended to forecast (predict) future values.

**Irregular Component:**

We want to increase our understanding of a time series by picking out itsmain features. One of these main features is the irregular component (or

‘noise’). Descriptive techniques may be extended to forecast (predict) futurevalues. The irregular component is that left over when the other components ofthe series (trend, seasonal and cyclical) have been accounted for.

**Exponential smoothing** is a widely used technique that can be applied to timeseries data, either to produce smoothed data for presentation, or to makeforecasts. The time series data themselves are a sequence of observations. Theobserved phenomenon may be an essentially random process, or it may be anorderly, but noisy, process. Whereas in the simple moving average the pastobservations are weighted equally, exponential smoothing assigns exponentiallydecreasing weights over time.

Exponential smoothing is commonly applied to financial market and economicdata, but it can be used with any discrete set of repeated measurements. Theraw data sequence is often represented by {xt}, and the output of theexponential smoothing algorithm is commonly written as {st} which may beregarded as our best estimate of what the next value of x will be. When thesequence of observations begins at time t = 0, the simplest form of exponentialsmoothing is given by the formulas

s0 = x0 st = αxt-1 + (1-α) st-1

Where a is the smoothing factor, and 0 < a < 1.

**The exponential moving average:**

The simplest form of exponential smoothing is given by the formulae s0 = x0 st = αxt-1 + (1- α) st-1 + α (xt-1 - st-1)

Where α is the smoothing factor, and 0 < a < 1. In other words, the smoothed statistic st is a simple weighted average of the previous observation xt-1 and the previous smoothed statistic st-1. Simple exponential smoothing is easily applied, and it produces a smoothed statistic as soon as two observations are available.

values of α close to one have less of a smoothing effect and give greater weight to recent changes in the data, while values of a closer to zero have a greater smoothing effect and are less responsive to recent changes. There is no formally correct procedure for choosing α. Sometimes the statistician's judgment is used to choose an appropriate factor. Alternatively, a statistical technique may be used to optimize the value of α for which the sum of the quantities (sn-1 – xn-1)2 is minimized.

The simple form of the exponential smoothing is also known as an

“exponentially weighted moving average”. Technically it can also be classified as an ARIMA (0,1,1) model with no constant term. **why is it "exponential"?**

By direct substitution of the defining equation for simple exponential smoothing back into itself we find that

st = αxt-1 + (1- α) st-1

= αxt-1 + α (1- α) xt-2 + (1-α) st-2

=α [xt-1 + (1-α) xt-2 + (1-α)2xt-3 + (1-α)3 xt-4 +………] +(1-α) t-1 x0.

In other words, as time passes the smoothed statistic st becomes the weighted average of a greater and greater number of the past observations xt-n, and the weights assigned to previous observations are in general proportional to the terms of the geometric progression {1, (1 – α), (1-α)2 (1-α)3….}. A geometric progression is the discrete version of an exponential function, so this is where the name for this smoothing method originated.

**2. Holt's Double exponential smoothing method for analysis and forecasting of time series:**

Holt’s method uses double exponential smoothing to smooth out noise and to forecast data that exhibit a trend. Holt's double exponential smoothing method calculate a level component and a trend component at each period and it uses two smoothing parameters for updating these components. Double exponential smoothing uses the level and trend components to generate forecasts. Initial values for these components are obtained either by back casting (if you use optimal weights), or from a linear regression on time (if you specify weights).

The first weight updates the level component. Theoretically, this can be any Number between 0 and 2, but it is normally a value between 0 and 1. The second weight updates the trend component. This can be any value between 0 and 100, although it is normally a value between 0 and 1.

**The Holt's model:**

Yt = Lt + Ttt + et each of the two parameters in this model requires an updating (smoothing) equation:

The updating equation for the LEVEL:

Lt = αYt + (1 - α) (Lt-1 + Tt-1)

and the TREND updating equation:

Tt = γ (Lt – Lt-1) +(1- γ) Tt-1

The fitted value or one-period-ahead forecast is given by

Ŷt = Lt-1 + Tt-1

where

* Lt is the level at time t
* α is the weight for the level
* Tt is the trend at time t
* γ is the weight for the trend
* Yt is the data value at time t
* Ŷt is the fitted value, or one-period-ahead forecast, at time t

**NOTE:** Initial values for the level and trend components are obtained from a linear regression on time.

**2.2 AUTO REGRESSIVE MOVING AVERAGE MODEL :**

Auto Regressive Moving Average (ARMA) is a combination of two fundamental time series models: Auto Regressive (AR) and Moving Average (MA). These models are used to describe and predict the behaviour of a time series based on its past values and, in the case of ARMA, on past errors (residuals) as well.

**1. Auto Regressive (AR) Model**:

An Auto Regressive model of order p, denoted as AR(p), predicts the current value of a time series based on its past p values. In other words, the current value is considered to be a linear combination of the previous p values, each multiplied by a coefficient. The AR model captures the autoregressive relationship within the time series, where the current value is influenced by its own past values.

Mathematically, an AR(p) model can be expressed as:

Y \_t = c + φ₁ \* y\_(t-1) + φ₂ \* y\_(t-2) + ... + φ\_p \* y\_(t-p) + ε\_t

Where:

- y\_t is the current value of the time series.

- c is a constant or intercept term.

- φ₁ to φ\_p are the autoregressive coefficients for lags 1 to p.

- ε\_t is the white noise or error term at time t.

**2. Moving Average (MA) Model**:

A Moving Average model of order q, denoted as MA(q), predicts the current value of a time series based on past q error terms (residuals). The idea is to capture the short-term fluctuations or noise in the time series. The MA model assumes that the current value of the series is related to a linear combination of the past q error terms.

Mathematically, an MA(q) model can be expressed as:

y\_t = μ + ε\_t + θ₁ \* ε\_(t-1) + θ₂ \* ε\_(t-2) + ... + θ\_q \* ε\_(t-q)

Where:

- y\_t is the current value of the time series.

- μ is the mean of the time series.

- ε\_t is the white noise or error term at time t.

- θ₁ to θ\_q are the moving average coefficients for lags 1 to q.

**3. ARMA Model**:

The ARMA model combines the AR and MA models to create a comprehensive model that captures both autoregressive relationships and short-term fluctuations. An ARMA(p, q) model includes both autoregressive terms and moving average terms to predict the current value of the time series.

Mathematically, an ARMA(p, q) model can be expressed as a combination of the AR and MA equations:

y\_t = c + φ₁ \* y\_(t-1) + φ₂ \* y\_(t-2) + ... + φ\_p \* y\_(t-p) + ε\_t + θ₁ \* ε\_(t-1) + θ₂ \* ε\_(t-2) + ... + θ\_q \* ε\_(t-q)

ARMA models are widely used for modeling time series data in various fields, including economics, finance, and engineering. However, it's important to note that ARMA models assume stationary data, meaning that the statistical properties of the data do not change over time. If the data is non-stationary, techniques like differencing can be applied to make it stationary before applying ARMA modeling.

**2.3 Autoregressive Integrated Moving Average (ARIMA) model:**

In time series analysis, **an autoregressive integrated moving average**

**(ARIMA)** model is a generalisation of an autoregressive moving average (ARMA) model. These models are fitted to time series data either to better understand the data or to predict future points in the series. They are applied in some cases where data show evidence of non-stationarity, where an initial differencing step (corresponding to the “integrated” part of the model) can be applied to remove the non-stationarity.

The model is generally referred to as an ARIMA (p, d, q) model where p, d, and g are integers greater than or equal to zero and refer to the order of the autoregressive, integrated, and moving average parts of the model respectively. ARIMA models form an important part of the Box-Jenkins approach to time-series modelling.

ARIMA models are, in theory, the most general class of models for forecasting a time series which can be stationarized by transformations such as differencing and lagging. In fact, the easiest way to think of ARIMA models is as fine-tuned versions of random-walk and random-trend models: the finetuning consists of adding lags of the differenced series and/or lags of the forecast errors to the prediction equation, as needed to remove any last traces of autocorrelation from the forecast errors.

The acronym ARIMA stands for "Auto-Regressive Integrated Moving Average." Lags of the differenced series appearing in the forecasting equation are called “auto-regressive” terms, lags of the forecast errors are called “moving average” terms, and a time series which needs to be differenced to be made stationary is said to be an “integrated" version of a stationary series. Random-walk and smoothing random-trend models, autoregressive models, and exponential smoothing models (i.e., exponential weighted moving averages) are all special cases of ARIMA models.

A non-seasonal ARIMA model is classified as an ARIMA (p, d, q) model, where:

* **p** is the number of autoregressive terms,
* **d** is the number of non-seasonal differences, and
* **q** is the number of lagged forecast errors in the prediction equation.

To identify the appropriate ARIMA model for a time series, we begin by identifying the order(s) of differencing needing to stationarize the series and remove the gross features of seasonality, perhaps in conjunction with a variance-stabilizing transformation such as logging or deflating. If we stop at this point and predict that the differenced series is constant, we have merely fitted a random walk or random trend model. (Recall that the random walk model predicts the first difference of the series to be constant, the seasonal random walk model predicts the seasonal difference to be constant, and the seasonal random trend model predicts the first difference of the seasonal difference to be constant--usually zero.) However, the best random walk or random trend model may still have auto correlated errors, suggesting that additional factors of some kind are needed in the prediction equation.

**ARIMA (0,1,0) = random walk:** In models we have studied previously, we have encountered two strategies for eliminating autocorrelation in forecast errors. One approach, which we first used in regression analysis, was the addition of lags of the stationarized series. For example, suppose we initially fit the random-walk-with-growth model to the time series Y. The prediction equation for this model can be written as:

**Ŷ(t) - Y(t-1) =** 𝝁

…where the constant term (here denoted by "mu") is the average difference in Y. This can be considered as a degenerate regression model in which DIFF(Y) is the dependent variable and there are no independent variables other than the constant term. Since it includes (only) a non-seasonal difference and a constant term, it is classified as an "ARIMA (0,1,0) model with constant.” Of course, the random walk without growth would be just an ARIMA (0,1,0) model without constant.

**ARIMA (1,1,0) = differenced first-order autoregressive model:** If the errors of the random walk model are autocorrelated, perhaps the problem can be fixed by adding one lag of the dependent variable to the prediction equation-- i.e., by regressing DIFF(Y) on itself lagged by one period. This would yield the following prediction equation:

**Ŷ(t) - Y(t-1) =** 𝝁 **+** ∅(𝒀(𝒕 − 𝟏) − 𝒀(𝒕 − 𝟐)

which can be rearranged to

**Ŷ(t) =** 𝝁 **+ Y (t-1) +** ∅(𝒀(𝒕 − 𝟏) − 𝒀(𝒕 − 𝟐)

This is a first-order autoregressive, or "AR (1)", model with one order of nonseasonal differencing and a constant term--i.e., an "ARIMA (1,1,0) model with constant." Here, the constant term is denoted by "mu" and the autoregressive coefficient is denoted by "phi”, in keeping with the terminology for ARIMA models popularized by Box and Jenkins. (In the output of the Forecasting procedure in Stat graphics, this coefficient is simply denoted as the AR (1) coefficient.)

**ARIMA (0,1,1) without constant = simple exponential smoothing:** Another strategy for correcting auto correlated errors in a random walk model is suggested by the simple exponential smoothing model. Recall that for some non-stationary time series (e.g., one that exhibits noisy fluctuations around a slowly- varying mean), the random walk model does not perform as well as a moving average of past values. In other words, rather than taking the most recent observation as the forecast of the next observation, it is better to use an average of the last few observations in order to filter out the noise and more accurately estimate the local mean. The simple exponential smoothing model uses an exponentially weighted moving average of past values to achieve this effect. The prediction equation for the simple exponential smoothing model can be written in a number of mathematically equivalent ways, one of which is:

**Ŷ(t) = Y(t-1) –** 𝜽 **e(t-1)**

...where e(t-1) denotes the error at period t-1. Note that this resembles the prediction equation for the ARIMA (1,1,0) model, except that instead of a multiple of the lagged difference it includes a multiple of the lagged forecast error. (It also does not Include a constant term--yet.) The coefficient of the lagged forecast error is denoted by the Greek letter 𝜃 (again following Box and Jenkins) and it is conventionally written with a negative sign for reasons of mathematical symmetry.

When a lagged forecast error is included in the prediction equation as shown above, it Is referred to as a "moving average" (MA) term. The simple exponential smoothing model is therefore a first-order moving average ("MA

(1)") model with one order of non-seasonal differencing and no constant term –i.e., an "ARIMA (0,1,1) model without constant". This means that in Statgraphics (or any other statistical software that supports ARIMA models) we can actually fit a simple exponential smoothing by specifying it as an ARIMA (0,1,1) model without constant.

**ARIMA (0,1,1) with constant = simple exponential smoothing with growth:** By implementing the SES model as an ARIMA model, we actually gain some flexibility. First of all, the estimated MA(1) coefficient is allowed to benegative: this corresponds to a smoothing factor larger than 1 in an SES model,which is usually not allowed by the SES model-fitting procedure. Second, wehave the option of including a constant term in the ARIMA model if we wish, inorder to estimate an average non-zero trend. The ARIMA (0,1,1) model withconstant has the prediction equation:

Ŷ(t) = 𝜇 + Y(t-1) – 𝜃 e(t-1)

The one-period-ahead forecasts from this model are qualitatively similar to those of the SES model, except that the trajectory of the long-term forecasts ls typically a sloping line (whose slope is equal to mu) rather than a horizontal line.

**ARIMA (0,2,1) or (0,2,2) without constant = linear exponential smoothing:** Linear exponential smoothing models are ARIMA models which use two non-seasonal differences in conjunction with MA terms. The second difference of a series Y is not simply the difference between Y and itself lagged by two periods, but rather It is the first difference of the first difference-i.e., the change-in-the-change of Y at period t. Thus, **the second difference of Y at period t is equal to (Y(t)-Y(t-1)) - (Y(t-1)-Y(t-2)) = Y(t) - 2Y(t-1) + Y(t-2).** A second difference of a discrete function is analogous to a second derivative of a continuous function: it measures the "acceleration" or "curvature" in the function at a given point in time.

The ARIMA (0,2,2) model without constant predicts that the second difference of the series equals a linear function of the last two forecast errors:

**t – 2Y(t-1) + Y(t-2) = -****e(t-1)-**

which can be rearranged as:

**t = 2Y(t-1) - Y(t-2) -** **e(t-1) -**

Where theta-1 and theta-2 are the MA (1) and MA (2) coefficients. This is essentially the same as Brown's linear exponential smoothing model, with the MA (1) coefficient corresponding to the quantity 2\*(1-alpha) in the LES model. To see this connection, recall that forecasting equation for the LES model is:

**t = 2Y(t-1) - Y(t-2) – 2(1-α) e(t-1) + (1-α)2**

Upon comparing terms, we see that the MA (1) coefficient corresponds to the quantity 2\*(1-alpha) and the MA (2) coefficient corresponds to the quantity (1-alpha) ^2 (i.e., " minus (1-alpha) squared"). If alpha is larger than 0.7, the corresponding MA (2) term would be less than 0.09, which might not be significantly different from zero, in which case an ARIMA (0,2,1) model probably would be identified.

**A "mixed" model--ARIMA (1,1,1):** The features of autoregressive and moving average models can be "mixed" in the same model. for example, an ARIMA (1,1,1) model with constant would have the prediction equation:

**Ŷ(t) =**  **+ Y (t-1) +** 

Normally, though, we will try to stick to "un mixed" models with either onlyAR or only-MA terms, because including both kinds of terms in the same modelsometimes leads to over-fitting of the data and non-uniqueness of thecoefficients.

The notation AR(p) refers to the autoregressive model of order p. The AR(P)model is written

**Xt** **.**

Where 𝜑𝑖, … … , 𝜑𝑝 are the **parameters** of the model, c is a constant and 𝜀t is white-noise. The constant term is omitted by many authors for simplicity.

The notation MA(q) refers to the moving average model of order q:

**Xt** 

where the 𝜃1, … … , 𝜃𝑞 are the parameters of the model, μ is the expectation of

Xt (often assumed to equal 0), and the ct , ct-1 are again, white-noise error terms The moving average model is essentially a finite impulse response filter with some additional Interpretation placed on It.

Use ARIMA to model time series behavior and to generate forecasts. ARIMA fits a Box-Jenkins ARIMA model to a time series. ARIMA stands for Autoregressive integrated Moving Average with each term representing steps taken in model construction until only random noise remains. ARIMA modelling differs from the other time series methods discussed in the chapter in the fact that ARIMA modelling uses correlation techniques. ARIMA can be used to model patterns that may not be visible in plotted data.

**Applications:**

ARIMA Is appropriate when a system Is a function of a series of unobserved shocks (the MA part) as well as its own behaviour. For example, stock prices may be shocked by fundamental information as well as exhibiting technical trending and mean-reversion effect due to market participants.

CHAPTER III

**BSE INDEX SENSEX AND NSE INDEX NIFTY – MODELING AND**

**FORCASTING USING TIME**

**R PROGRAMMING**

**R** is a [programming language](https://en.wikipedia.org/wiki/Programming_language) for [statistical computing](https://en.wikipedia.org/wiki/Statistical_computing) and graphics supported by the R Core Team and the R Foundation for Statistical Computing. Created by statisticians [Ross Ihaka](https://en.wikipedia.org/wiki/Ross_Ihaka) and [Robert Gentleman,](https://en.wikipedia.org/wiki/Robert_Gentleman_(statistician)) R is used among [data miners,](https://en.wikipedia.org/wiki/Data_mining) [bioinformaticians](https://en.wikipedia.org/wiki/Bioinformatics) and [statisticians](https://en.wikipedia.org/wiki/Statistician) for [data analysis](https://en.wikipedia.org/wiki/Data_analysis) and developing [statistical software.](https://en.wikipedia.org/wiki/Statistical_software) Users have created packages to augment the functions of the R language.

According to user surveys and studies of scholarly literature databases, R is one of the most commonly used programming languages in data mining.[[8]](https://en.wikipedia.org/wiki/R_(programming_language)#cite_note-8) As of April 2023, R ranks 16th in the [TIOBE index,](https://en.wikipedia.org/wiki/TIOBE_index) a measure of programming language popularity, in which the language peaked in 8th place in August 2020.

The official R software environment is an open-source [free software](https://en.wikipedia.org/wiki/Free_software) environment within the [GNU package,](https://en.wikipedia.org/wiki/List_of_GNU_packages) available under the [GNU General Public License.](https://en.wikipedia.org/wiki/GNU_General_Public_License) It is written primarily in [C,](https://en.wikipedia.org/wiki/C_(programming_language)) [Fortran,](https://en.wikipedia.org/wiki/Fortran) and R itself (partially [self-hosting)](https://en.wikipedia.org/wiki/Self-hosting_(compilers)). Precompiled [executables](https://en.wikipedia.org/wiki/Executable) are provided for various [operating systems.](https://en.wikipedia.org/wiki/Operating_system) R has a [command line interface.](https://en.wikipedia.org/wiki/Command_line_interface) Multiple third-party [graphical user interfaces](https://en.wikipedia.org/wiki/Graphical_user_interface) are also available, such as [RStudio,](https://en.wikipedia.org/wiki/RStudio) an [integrated development environment,](https://en.wikipedia.org/wiki/Integrated_development_environment) and [Jupyter,](https://en.wikipedia.org/wiki/Jupyter) a [notebook interface.](https://en.wikipedia.org/wiki/Notebook_interface)

R was started by professors [Ross Ihaka](https://en.wikipedia.org/wiki/Ross_Ihaka) and [Robert Gentleman](https://en.wikipedia.org/wiki/Robert_Gentleman_(statistician)) as a programming language to teach introductory statistics at the [University of Auckland.](https://en.wikipedia.org/wiki/University_of_Auckland) The language took heavy inspiration from the [S programming language](https://en.wikipedia.org/wiki/S_(programming_language)) with most S programs able to run unaltered in R as well as from [Scheme's](https://en.wikipedia.org/wiki/Scheme_(programming_language)) [lexical scoping](https://en.wikipedia.org/wiki/Lexical_scoping) allowing for local variables. The name of the language comes from being an S language successor and the shared first letter of the authors, Ross and Robert. Ihaka and Gentleman first shared [binaries](https://en.wikipedia.org/wiki/Binary_file) of R on the data archive StatLib and the *s-news* mailing list in August 1993. In June 1995, statistician Martin Machler convinced Ihaka and Gentleman to make R [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source_software) under the [GNU General Public License.](https://en.wikipedia.org/wiki/GNU_General_Public_License) Mailing lists for the R project began on 1 April 1997 preceding the release of version 0.50. R officially became a GNU project on 5 December 1997 when version 0.60 released. The first official 1.0 version was released on 29 February 2000.

The [Comprehensive R Archive Network](https://en.wikipedia.org/wiki/R_package#Comprehensive_R_Archive_Network_(CRAN)) (CRAN) was founded in 1997 by Kurt Hornik and Fritz Leisch to host R's [source code,](https://en.wikipedia.org/wiki/Source_code) executable files, documentation, and user-created packages. Its name and scope mimics the [Comprehensive TeX Archive Network](https://en.wikipedia.org/wiki/Comprehensive_TeX_Archive_Network) and the [Comprehensive Perl Archive Network.](https://en.wikipedia.org/wiki/Comprehensive_Perl_Archive_Network) CRAN originally had three mirrors and 12 contributed packages. As of December 2022, it has 103 mirrors and 18,976 contributed packages.

The R Core Team was formed in 1997 to further develop the language. As of January 2022, it consists of Chambers, Gentleman, Ihaka, and Machler, plus statisticians Douglas Bates, [Peter Dalgaard,](https://en.wikipedia.org/wiki/Peter_Dalgaard) [Kurt Hornik,](https://en.wikipedia.org/w/index.php?title=Kurt_Hornik&action=edit&redlink=1) Michael Lawrence, Friedrich Leisch, Uwe Ligges, [Thomas Lumley,](https://en.wikipedia.org/wiki/Thomas_Lumley_(statistician)) Sebastian Meyer, Paul Murrell, Martyn Plummer, [Brian Ripley,](https://en.wikipedia.org/wiki/Brian_Ripley) Deepayan Sarkar, Duncan Temple Lang, [Luke Tierney,](https://en.wikipedia.org/wiki/Luke_Tierney) and Simon Urbanek, as well as computer scientist Tomas Kalibera. Stefano Iacus, Guido Masarotto, Heiner Schwarte, Seth Falcon, Martin Morgan, and Duncan Murdoch were members. In April 2003, the R Foundation was founded as a non-profit organization to provide further support for the R project.

Programming

R is an [interpreted language;](https://en.wikipedia.org/wiki/Interpreted_language) users can access it through a [command-line interpreter.](https://en.wikipedia.org/wiki/Command-line_interpreter) If a user types 2+2 at the R command prompt and presses enter, the computer replies with 4.

Programming R to Create a Bar Graph

R supports [procedural programming](https://en.wikipedia.org/wiki/Procedural_programming) with [functions](https://en.wikipedia.org/wiki/Function_(computer_science)) and, for some functions, [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming) with [generic functions.](https://en.wikipedia.org/wiki/Generic_function) Due to its  [S](https://en.wikipedia.org/wiki/S_(programming_language)) heritage, R has stronger [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming) facilities than most statistical computing languages. Extending it is facilitated by its [lexical scoping](https://en.wikipedia.org/wiki/Lexical_scoping) rules, which are derived from Scheme. R uses [S](https://en.wikipedia.org/wiki/S_(programming_language)) syntax (not to be confused with [S-expressions)](https://en.wikipedia.org/wiki/S-expressions) to represent both data and code . R's extensible object system includes objects for (among others): [regression models,](https://en.wikipedia.org/wiki/Regression_analysis) [timeseries](https://en.wikipedia.org/wiki/Time-series) and [geo-](https://en.wikipedia.org/wiki/Spatial_analysis)spatial coordinate[s.](https://en.wikipedia.org/wiki/Spatial_analysis) Advanced users can write C,

C++, [Java,](https://en.wikipedia.org/wiki/Java_(programming_language)) [.NET](https://en.wikipedia.org/wiki/.NET_Framework) or [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) code to manipulate R objects directly.

|  |
| --- |
| print |

|  |
| --- |
| print(object name) |

Functions are [first-class](https://en.wikipedia.org/wiki/First-class_functions) objects and can be manipulated in the same way as data objects, facilitating [meta-programming](https://en.wikipedia.org/wiki/Meta-programming) that allows [multiple dispatch.](https://en.wikipedia.org/wiki/Multiple_dispatch) Function arguments are passed by value, and are [lazy](https://en.wikipedia.org/wiki/Lazy_evaluation) that is to say, they are only evaluated when they are used, not when the function is called. A generic function acts differently depending on the [classes](https://en.wikipedia.org/wiki/Class_(computer_programming)) of the arguments passed to it. In other words, the generic function [dispatches](https://en.wikipedia.org/wiki/Dynamic_dispatch) the [method](https://en.wikipedia.org/wiki/Method_(computer_science)) implementation specific to that object's [class.](https://en.wikipedia.org/wiki/Class_(computer_programming)) For example, R has a [generic](https://en.wikipedia.org/wiki/Generic_function)  function that can print almost every [class](https://en.wikipedia.org/wiki/Class_(computer_programming)) of [object](https://en.wikipedia.org/wiki/Object_(computer_science)) in R with . R is highly extensible through the use of packages for specific functions and specific applications.

Basic syntax

The following examples illustrate the basic [syntax of the language](https://en.wikipedia.org/wiki/Programming_language_syntax) and use of the command-line interface. (An expanded list of standard language features can be found in the R manual, "An Introduction to R".

|  |
| --- |
| <- |

In R, the generally preferred [assignment operator](https://en.wikipedia.org/wiki/Assignment_(computer_science)) is an arrow made from two characters , although = can be used in some cases

**>** x <- 1:6 *# Create a numeric vector in the current environment* **>** y <- x^2 *# Create vector based on the values in x.*

**>** print(y)

Structure of a function

One of R's strengths is the ease of creating new functions. Objects in the function body remain local to the function, and any data type may be returned. Example:

*# Declare function “f” with parameters “x”, “y “* *# that returns a linear combination of x and y.*

f <- function (x, y) { z <- 3 \* x + 4 \* y return(z) *## the return () function is optional here* }

Modelling and plotting

The R language has built-in support for data modelling and graphics. The following example shows how R can easily generate and plot a linear model with residuals.

|  |
| --- |
|  |

Diagnostic plots from plotting “model” (q.v. “plot. lm()” function). Notice the mathematical notation allowed in labels (lower left plot).

**>** x <- 1:6 *# Create x and y values*

**>** y <- x^2

**>** model <- lm(y ~ x) *# Linear regression model y = A + B \* x.*

**>** summary(model) # *Display an in-depth summary of the model.*

**DATA COLLECTION:**

We have collected the monthly SENSEX indices for the period2013 –

2022 and the weekly SENSEX indices for the years 2019 – 2022 on

BOMBAY STOCK EXCHANGE (BSE) from the website of BSE. We present the monthly data TABLE 1 and weekly data TABLE 2, which are given below.

Similarly, we have collected the monthly NIFTY indices for the period

2013 – 2022 and weekly NIFTY indices for the 2019 – 2022 on

NATIONAL STOCK EXCHANGE (NSE) from the website of NSE and are presented below in table 3 and table 4.

TABLE 1 : MONTHLY SENSEX INDICES OF BSE FOR

THE PERIOD JAN 2013 – DEC 2022

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Month | 2013 | 2014 | 2015 | 2016 | 2017 | 2018 | 2019 | 2020 | 2021 | 2022 |
| Jan | 19894 | 20513 | 29182 | 24826 | 27655 | 35965 | 36256 | 40723 | 46285 | 58014 |
| Feb | 18861 | 21120 | 29361 | 23002 | 28743 | 34184 | 35867 | 38297 | 49099 | 56247 |
| Mar | 18835 | 22386 | 27957 | 25341 | 29620 | 32968 | 38672 | 29468 | 49509 | 58568 |
| Apr | 19504 | 22417 | 27011 | 25606 | 29918 | 35168 | 39031 | 33717 | 48782 | 57060 |
| May | 19760 | 24217 | 27828 | 26667 | 31145 | 35322 | 39714 | 32424 | 51937 | 55566 |
| Jun | 19395 | 25413 | 27780 | 26999 | 30921 | 35423 | 39394 | 34915 | 52482 | 53018 |
| Jul | 19345 | 25894 | 28114 | 28051 | 32514 | 37606 | 37481 | 37606 | 52586 | 57570 |
| Aug | 18619 | 26638 | 26283 | 28452 | 31730 | 38645 | 37332 | 38628 | 57552 | 59537 |
| Sep | 19379 | 26630 | 26154 | 27865 | 31283 | 36227 | 38667 | 38067 | 59126 | 57426 |
| Oct | 21164 | 27865 | 26656 | 27930 | 33213 | 34442 | 40129 | 39614 | 59306 | 60746 |
| Nov | 20791 | 28693 | 26145 | 26652 | 33149 | 36194 | 40793 | 44149 | 57064 | 63099 |
| Dec | 21170 | 27499 | 26117 | 26626 | 34056 | 36068 | 41253 | 47751 | 58253 | 60840 |

TABLE 2: WEEKLY SENSEX INDICES OF BSE FOR THE YEARS 2019 – 2022

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| week | 2019 | 2020 | 2021 | 2022 |  | 2019 | 2020 | 2021 | 2022 |
| 1 | 35850 | 41859 | 48093 | 59601 | 27 | 38720 | 36693 | 52568 | 54178 |
| 2 | 35853 | 41528 | 49584 | 61235 | 28 | 38896 | 37418 | 53158 | 53416 |
| 3 | 36578 | 41155 | 49624 | 59464 | 29 | 38031 | 37434 | 52837 | 55681 |
| 4 | 35656 | 39872 | 46874 | 57276 | 30 | 37686 | 36939 | 52653 | 56857 |
| 5 | 36582 | 40979 | 50614 | 58788 | 31 | 36699 | 38182 | 54492 | 58298 |
| 6 | 36395 | 41055 | 51531 | 58926 | 32 | 37581 | 38050 | 54843 | 59332 |
| 7 | 35498 | 40363 | 51324 | 57892 | 33 | 37402 | 38799 | 55629 | 60298 |
| 8 | 36213 | 38144 | 51039 | 54529 | 34 | 37494 | 38628 | 55949 | 58774 |
| 9 | 36063 | 35634 | 50846 | 55102 | 35 | 37332 | 38417 | 57852 | 58766 |
| 10 | 37054 | 31390 | 51279 | 55464 | 36 | 37145 | 38756 | 58305 | 59688 |
| 11 | 38095 | 25981 | 49216 | 57863 | 37 | 37123 | 38034 | 59141 | 59934 |
| 12 | 37808 | 28440 | 48440 | 57595 | 38 | 39090 | 37981 | 59885 | 59119 |
| 13 | 38871 | 27590 | 50029 | 58568 | 39 | 38667 | 38973 | 59126 | 56409 |
| 14 | 38700 | 30690 | 49746 | 59034 | 40 | 37531 | 40593 | 59677 | 58222 |
| 15 | 38905 | 31648 | 40083 | 58338 | 41 | 38214 | 40431 | 61305 | 57235 |
| 16 | 38645 | 31743 | 48080 | 57911 | 42 | 39298 | 40145 | 60923 | 59202 |
| 17 | 39067 | 31715 | 49765 | 57521 | 43 | 39058 | 39757 | 59984 | 59756 |
| 18 | 38600 | 31561 | 48949 | 55702 | 44 | 40301 | 42597 | 60067 | 60836 |
| 19 | 37090 | 30028 | 48690 | 52930 | 45 | 40345 | 43443 | 59919 | 60613 |
| 20 | 39352 | 30672 | 49564 | 52792 | 46 | 40284 | 44077 | 59636 | 61750 |
| 21 | 39683 | 33303 | 51115 | 54252 | 47 | 40889 | 44149 | 58795 | 62272 |
| 22 | 40267 | 34370 | 52232 | 58189 | 48 | 40802 | 45426 | 58461 | 63284 |
| 23 | 39784 | 33228 | 52300 | 55320 | 49 | 40487 | 46253 | 58807 | 62570 |
| 24 | 38960 | 34911 | 52323 | 51495 | 50 | 40938 | 45553 | 57901 | 61799 |
| 25 | 39122 | 34961 | 52699 | 52265 | 51 | 41642 | 47353 | 57315 | 60826 |
| 26 | 39686 | 36487 | 52318 | 53018 | 52 | 41558 | 47751 | 57794 | 61133 |

2. DATA ANALYSIS :

In time series analysis there are many methods for handling, but here we adopt the following two methods for analysis and forecasting of the SENSEX indices and NIFTY indices.

1. moving average
2. HOLT’S single and double exponential method.
3. ARIMA method.

These methods are already explained elaborately in Chapter

– II. Further,the for selection of these methods is also explained in Chapter – I.

In this section,we carry out the time series analysis for the following stock market index data:

* Monthly SENSEX indices (table 1)
* Weekly SENSEX indices (table 2)
* Monthly NIFTY indices(table 3)
* Weekly NIFTY indices (table 4)

The sequence of various steps,those we have followed for the analysis of each of the above indices data,is as follows:

1. Plotting of time series graph
2. Analysis using moving average , single and double exponential method.
3. Modelling using of ARIMA method.
4. Comparison of single and double exponential and ARIMA models for selection of better model.

IN THE NEXT SECTION ,using the better model, we compute the forecasted monthly indices for the years 2013 – 2022 and forecasted weekly indices for the period 2019 – 2022.

**As** a first step , using R,we draw the time series plot for the BSE monthly SENSEX indices for the period **JAN 2019 – DEC 2022** and presented below.

**S**econd step enter the data in the excel and then write the R code.

**WE CARRY OUT THE ABOVE STEPS OF THE TIME SERIES ANALYSIS USING R**

**I . Analysis of monthly SENSEX indices of BSE :**

1.MOVING AVERAGE

CODE :

**library(forecast)**

**library(tseries)**

**library(ggplot2)**

**# Create a data frame from the provided data**

**data <- data.frame(**

**Month = seq.Date(as.Date("2013-01-01"), as.Date("2022-12-01"), by = "months"),**

**Value = c(**

**19894, 20513, 29182, 24826, 27655, 35965, 36256, 40723, 46285, 58014,**

**18861, 21120, 29361, 23002, 28743, 34184, 35867, 38297, 49099, 56247,**

**18835, 22386, 27957, 25341, 29620, 32968, 38672, 29468, 49509, 58568,**

**19504, 22417, 27011, 25606, 29918, 35168, 39031, 33717, 48782, 57060,**

**19760, 24217, 27828, 26667, 31145, 35322, 39714, 32424, 51937, 55566,**

**19395, 25413, 27780, 26999, 30921, 35423, 39394, 34915, 52482, 53018,**

**19345, 25894, 28114, 28051, 32514, 37606, 37481, 37606, 52586, 57570,**

**18619, 26638, 26283, 28452, 31730, 38645, 37332, 38628, 57552, 59537,**

**19379, 26630, 26154, 27865, 31283, 36227, 38667, 38067, 59126, 57426,**

**21164, 27865, 26656, 27930, 33213, 34442, 40129, 39614, 59306, 60746,**

**20791, 28693, 26145, 26652, 33149, 36194, 40793, 44149, 57064, 63099,**

**21170, 27499, 26117, 26626, 34056, 36068, 41253, 47751, 58253, 60840**

**)**

**)**

**# Set the data as a time series**

**ts\_data <- ts(data$Value, frequency = 12, start = c(2013, 1))**

**# Moving Average Method**

**ma\_model <- ma(ts\_data, order = 12)**

**ma\_forecast <- forecast(ma\_model, h = 24)**

**plot(ma\_forecast)**

**# Print forecasts**

**print("Moving Average Forecast:")**

**print(ma\_forecast)**

**OUTPUT:**

**Point Forecast Lo 80 Hi 80 Lo 95 Hi 95**

**Jul 2022 40954.07 39326.47 42581.66 38464.88 43443.25**

**Aug 2022 40954.07 38651.86 43256.27 37433.15 44474.98**

**Sep 2022 40954.07 38133.83 43774.31 36640.88 45267.25**

**Oct 2022 40954.07 37696.78 44211.35 35972.47 45935.66**

**Nov 2022 40954.07 37311.45 44596.68 35383.17 46524.97**

**Dec 2022 40954.07 36962.83 44945.30 34850.00 47058.13**

**Jan 2023 40954.07 36642.02 45266.12 34359.35 47548.78**

**Feb 2023 40954.07 36343.19 45564.94 33902.34 48005.79**

**Mar 2023 40954.07 36062.32 45845.81 33472.79 48435.34**

**Apr 2023 40954.07 35796.48 46111.65 33066.22 48841.91**

**May 2023 40954.07 35543.45 46364.68 32679.24 49228.89**

**Jun 2023 40954.07 35301.50 46606.63 32309.22 49598.91**

**Jul 2023 40954.07 35069.28 46838.85 31954.06 49954.07**

**Aug 2023 40954.07 34845.67 47062.46 31612.08 50296.05**

**Sep 2023 40954.07 34629.75 47278.38 31281.87 50626.27**

**Oct 2023 40954.07 34420.77 47487.36 30962.26 50945.87**

**Nov 2023 40954.07 34218.08 47690.05 30652.27 51255.86**

**Dec 2023 40954.07 34021.13 47887.00 30351.05 51557.08**

**Jan 2024 40954.07 33829.44 48078.70 30057.88 51850.25**

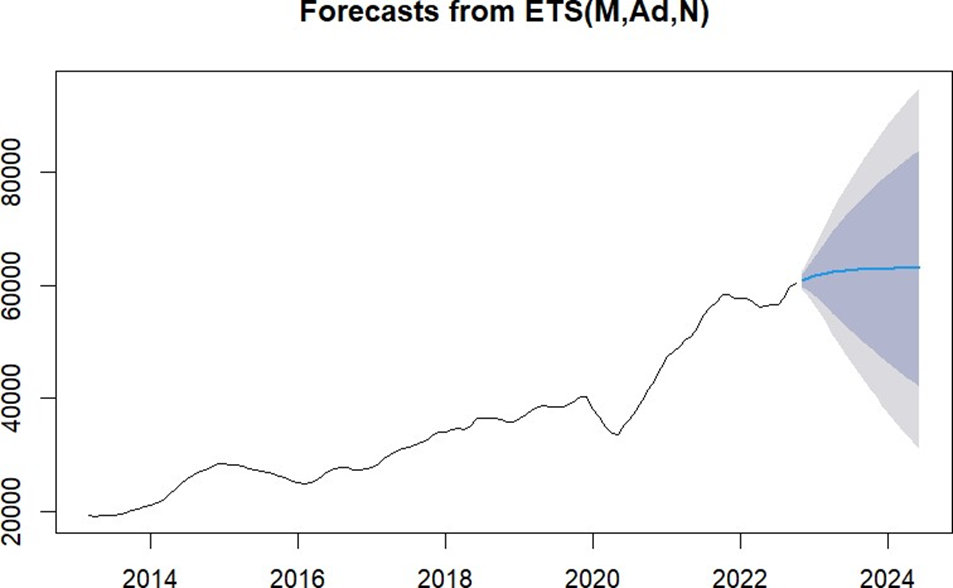
**Feb 2024 40954.07 33642.59 48265.54 29772.13 52136.00**

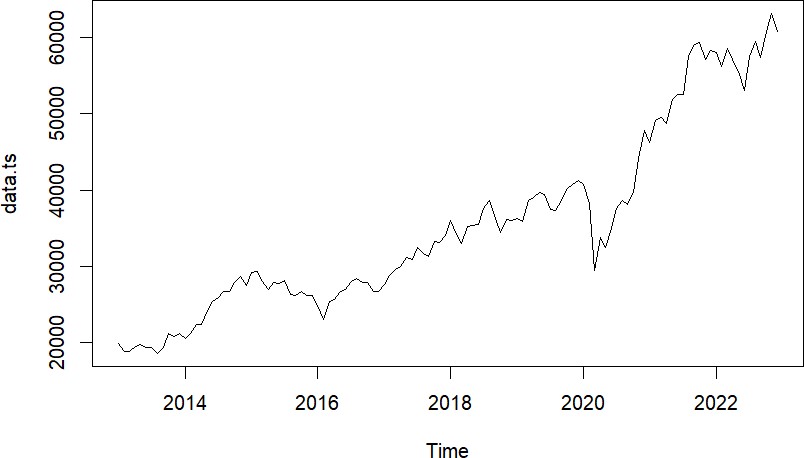
**Mar 2024 40954.07 33460.23 48447.90 29493.23 52414.90**

**Apr 2024 40954.07 33282.03 48626.10 29220.70 52687.43**

**May 2024 40954.07 33107.71 48800.42 28954.10 52954.03**

**Jun 2024 40954.07 32937. 48971.11 28693.06 53215.07**





2. Holt’s Single and double exponential CODE:

**library(forecast)**

**library(tseries)**

**library(ggplot2)**

**# Create a data frame from the provided data**

**data <- data.frame(**

**Month = seq.Date(as.Date("2013-01-01"), as.Date("2022-12-01"), by = "months"),**

**Value = c(**

**19894, 20513, 29182, 24826, 27655, 35965, 36256, 40723, 46285, 58014,**

**18861, 21120, 29361, 23002, 28743, 34184, 35867, 38297, 49099, 56247,**

**18835, 22386, 27957, 25341, 29620, 32968, 38672, 29468, 49509, 58568,**

**19504, 22417, 27011, 25606, 29918, 35168, 39031, 33717, 48782, 57060,**

**19760, 24217, 27828, 26667, 31145, 35322, 39714, 32424, 51937, 55566,**

**19395, 25413, 27780, 26999, 30921, 35423, 39394, 34915, 52482, 53018,**

**19345, 25894, 28114, 28051, 32514, 37606, 37481, 37606, 52586, 57570,**

**18619, 26638, 26283, 28452, 31730, 38645, 37332, 38628, 57552, 59537,**

**19379, 26630, 26154, 27865, 31283, 36227, 38667, 38067, 59126, 57426,**

**21164, 27865, 26656, 27930, 33213, 34442, 40129, 39614, 59306, 60746,**

**20791, 28693, 26145, 26652, 33149, 36194, 40793, 44149, 57064, 63099,**

**21170, 27499, 26117, 26626, 34056, 36068, 41253, 47751, 58253, 60840**

**)**

**)**

**# Set the data as a time series**

**ts\_data <- ts(data$Value, frequency = 12, start = c(2013, 1))**

**# Single Exponential Smoothing**

**ses\_model <- HoltWinters(ts\_data, beta = FALSE, gamma = FALSE)**

**ses\_forecast <- forecast(ses\_model, h = 24)**

**plot(ses\_forecast)**

**# Double Exponential Smoothing**

**des\_model <- HoltWinters(ts\_data)**

**des\_forecast <- forecast(des\_model, h = 24)**

**plot(des\_forest)**

**print("Single Exponential Smoothing Forecast:")**

**print(ses\_forecast)**

**print("Double Exponential Smoothing Forecast:")**

**print(des\_forecast)**

**OUTPUT :**

class(y$Sales)

[1] "integer"

> y1=as.numeric(y$Sales)

> d.y=diff(y2)

> summary(y1)

Min. 1st Qu. Median Mean 3rd Qu. Max.

18619 26649 33059 35370 39818 63099

> y2=na.omit(y1)

> summary(y2)

Min. 1st Qu. Median Mean 3rd Qu. Max.

18619 26649 33059 35370 39818 63099 > y3=ts(y2)

> library(forecast)

> plot(y3)

> model1=forecast::ets(y3,'ANN')

> forc1=forecast(model1,8)

> forc1

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

1. 60871.26 58561.34 63181.18 57338.54 64403.97
2. 60871.26 57627.40 64115.12 55910.21 65832.31
3. 60871.26 56907.74 64834.78 54809.58 66932.94
4. 60871.26 56300.00 65442.51 53880.13 67862.39
5. 60871.26 55764.08 65978.44 53060.50 68682.01
6. 60871.26 55279.29 66463.23 52319.08 69423.44
7. 60871.26 54833.29 66909.22 51636.99 70105.53 128 60871.26 54418.05 67324.47 51001.93 70740.59

> plot(forc1)

> forecast::accuracy(model1)

ME RMSE MAE MPE MAPE MASE ACF1

Training set 347.7672 1787.356 1277.141 0.8297336 3.580231 0.9943195 -0.04073729

> mod2=forecast::ets(y3,'AAN')

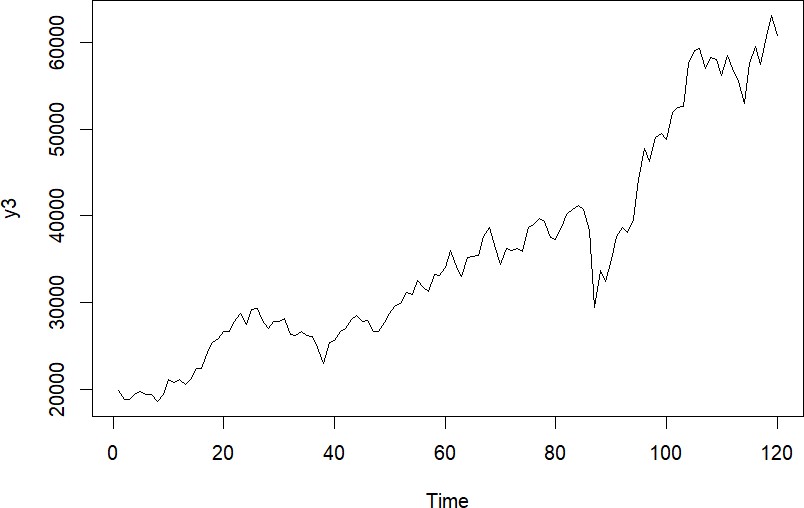
> forc2=forecast(mod2,8)

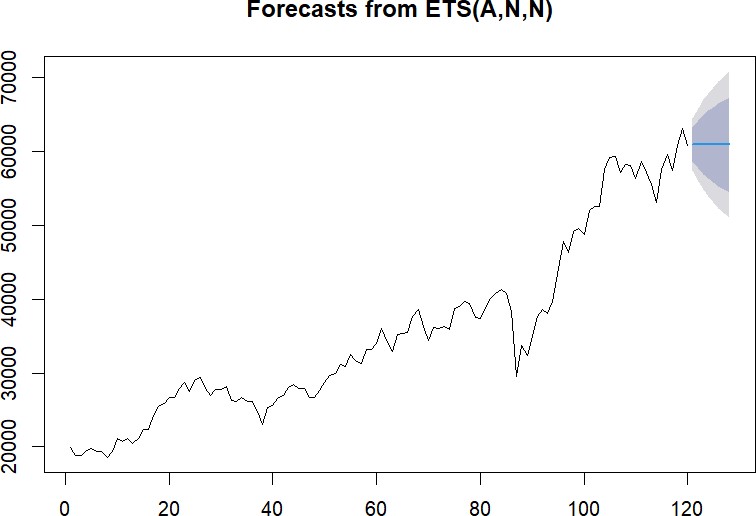
> plot(forc2)

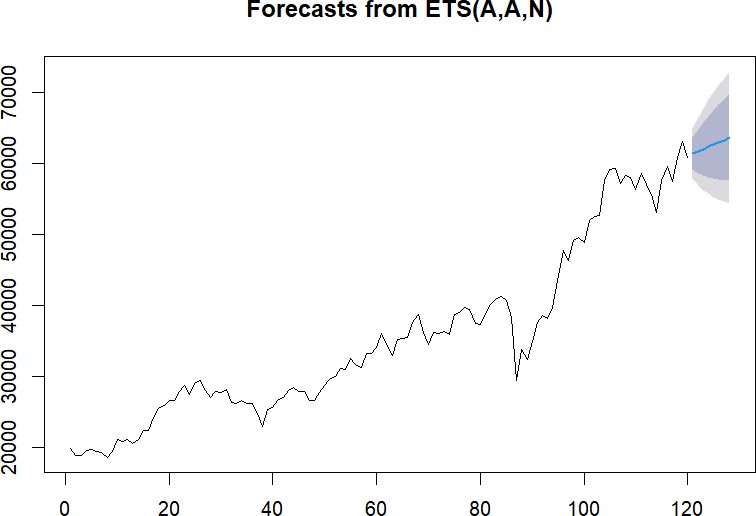
> forecast::accuracy(mod2)

ME RMSE MAE MPE MAPE MASE ACF1

Training set 33.00061 1751.853 1248.9 -0.1655884 3.528897 0.9723324 0.005531891







3 . ARIMA

CODE : # Create a data frame with the provided data

data.df <- data.frame(

Month = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"),

`2013` = c(19894, 20513, 29182, 24826, 27655, 35965, 36256, 40723, 46285, 58014),

`2014` = c(18861, 21120, 29361, 23002, 28743, 34184, 35867, 38297, 49099, 56247),

`2015` = c(18835, 22386, 27957, 25341, 29620, 32968, 38672, 29468, 49509, 58568),

`2016` = c(19504, 22417, 27011, 25606, 29918, 35168, 39031, 33717, 48782, 57060),

`2017` = c(19760, 24217, 27828, 26667, 31145, 35322, 39714, 32424, 51937, 55566),

`2018` = c(19395, 25413, 27780, 26999, 30921, 35423, 39394, 34915, 52482, 53018),

`2019` = c(19345, 25894, 28114, 28051, 32514, 37606, 37481, 37606, 52586, 57570),

`2020` = c(18619, 26638, 26283, 28452, 31730, 38645, 37332, 38628, 57552, 59537),

`2021` = c(19379, 26630, 26154, 27865, 31283, 36227, 38667, 38067, 59126, 57426),

`2022` = c(21164, 27865, 26656, 27930, 33213, 34442, 40129, 39614, 59306, 60746),

`2023` = c(20791, 28693, 26145, 26652, 33149, 36194, 40793, 44149, 57064, 63099),

`2024` = c(21170, 27499, 26117, 26626, 34056, 36068, 41253, 47751, 58253, 60840)

)

# Convert data to time series

y <- ts(data.df[, -1], start = c(2013, 1), frequency = 12)

# Calculate differences

d.y <- diff(y)

# Perform ADF tests

library(tseries)

adf\_test\_stationary <- adf.test(y[, 1], alternative = "stationary", k = 0)

adf\_test\_explosive <- adf.test(y[, 1], alternative = "explosive", k = 0)

adf\_test\_d.y <- adf.test(d.y[, 1], k = 0)

adf\_test\_d.y\_full <- adf.test(d.y[, 1])

# Perform ARIMA modeling

arima\_model <- arima(y[, 1], order = c(1, 0, 1))

# Predictions

n\_ahead <- 12

arima\_pred <- predict(arima\_model, n.ahead = n\_ahead)

# Plot data and differences

plot(y[, 1], main = "Original Data")

plot(d.y[, 1], main = "Differences")

# ADF test results

print("ADF Test - Stationary Hypothesis:")

print(adf\_test\_stationary)

print("ADF Test - Explosive Hypothesis:")

print(adf\_test\_explosive)

print("ADF Test for Differences:")

print(adf\_test\_d.y)

print("ADF Test for Differences (Full):")

print(adf\_test\_d.y\_full)

# ACF and PACF plots

par(mfrow = c(2, 2))

acf(y[, 1], main = "ACF - Original Data")

pacf(y[, 1], main = "PACF - Original Data")

acf(d.y[, 1], main = "ACF - Differences")

pacf(d.y[, 1], main = "PACF - Differences")

# ARIMA model summary

print("ARIMA Model Summary:")

print(summary(arima\_model))

**output:**

library(tseries)

> attach(data.df)

The following object is masked \_by\_ .GlobalEnv:

Sales

> y=sales

> y1=as.numeric(y)

> d.y=diff(y2)

> summary(y1)

Min. 1st Qu. Median Mean 3rd Qu. Max.

7610 11414 14137 14495 17091 123048

> y2=na.omit(y1)

> summary(y2)

Min. 1st Qu. Median Mean 3rd Qu. Max.

7610 11414 14137 14495 17091 123048

> plot(d.y)

> adf.test(y2,alternative = "stationary",k=0) Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -13.814, Lag order = 0, p-value = 0.01 alternative hypothesis: stationary

> adf.test(y2,alternative = "explosive",k=0) Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -13.814, Lag order = 0, p-value = 0.99 alternative hypothesis: explosive

>

> arima(y2,order = c(1,0,0)) Call:

arima(x = y2, order = c(1, 0, 0))

Coefficients: ar1 intercept 0.0937 14495.2462

s.e. 0.0691 619.4745 sigma^2 estimated as 65310682: log likelihood = -2156.17, aic = 4318.35

>

> adf.test(d.y,k=0)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -24.585, Lag order = 0, p-value = 0.01 alternative hypothesis: stationary

> adf.test(d.y)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -9.6904, Lag order = 5, p-value = 0.01

alternative hypothesis: stationary

> acf(y2)

> pacf(y2)

> acf(d.y)

> pacf(d.y)

> arima(y2,order = c(1,0,1))

Call:

arima(x = y2, order = c(1, 0, 1))

Coefficients:

ar1 ma1 intercept

0.9575 -0.8961 14501.686

s.e. 0.0561 0.0860 1267.794 sigma^2 estimated as 63168855: log likelihood = -2152.82, aic = 4313.64

> mydata.arima101=arima(y2,order = c(1,0,1))

> mydata.pred1=predict(mydata.arima101,n.ahead=100)

> mydata.pred1$pred

Time Series:

Start = 208

End = 307

Frequency = 1

[1] 16530.05 16443.86 16361.33 16282.31 16206.65 16134.21 16064.84 15998.42 15934.82 15873.93 [11] 15815.62 15759.79 15706.33 15655.14 15606.13 15559.20 15514.27 15471.24 15430.04 15390.60

[21] 15352.83 15316.66 15282.03 15248.87 15217.13 15186.73 15157.62 15129.75 15103.06 15077.51

[31] 15053.04 15029.61 15007.18 14985.70 14965.13 14945.44 14926.59 14908.53 14891.24 14874.69

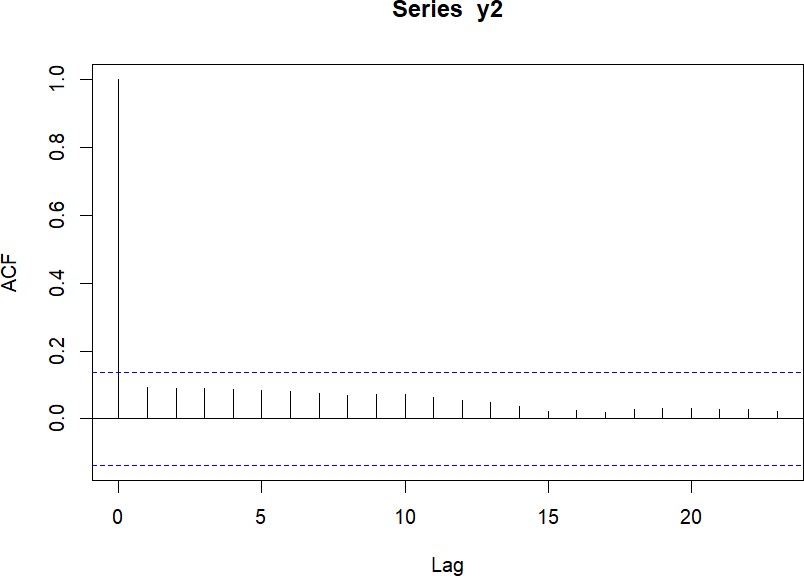
[41] 14858.84 14843.67 14829.13 14815.22 14801.90 14789.14 14776.93 14765.23 14754.03 14743.31

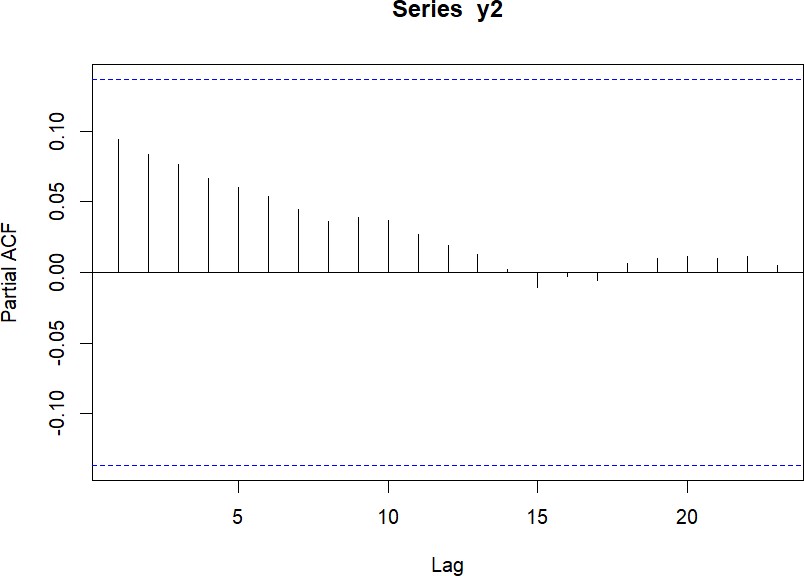
[51] 14733.04 14723.21 14713.80 14704.79 14696.16 14687.89 14679.98 14672.41 14665.15 14658.21

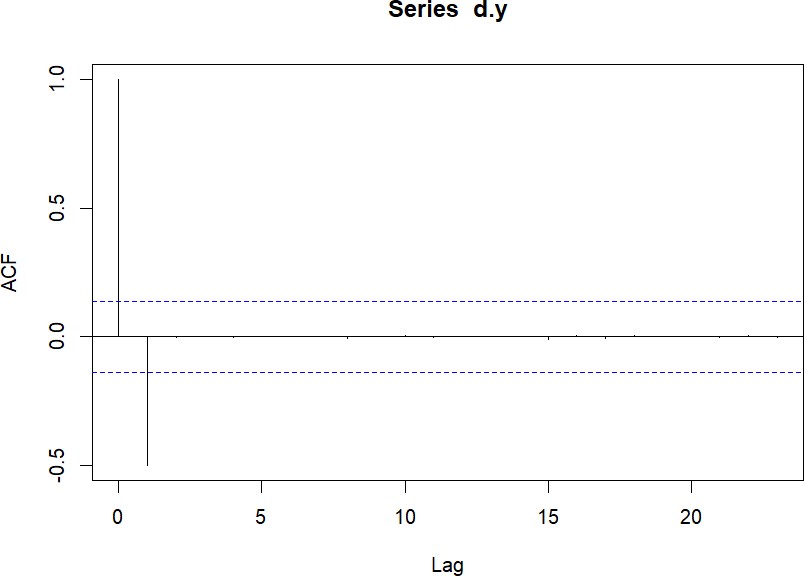
[61] 14651.56 14645.19 14639.09 14633.25 14627.66 14622.31 14617.18 14612.28 14607.58 14603.08

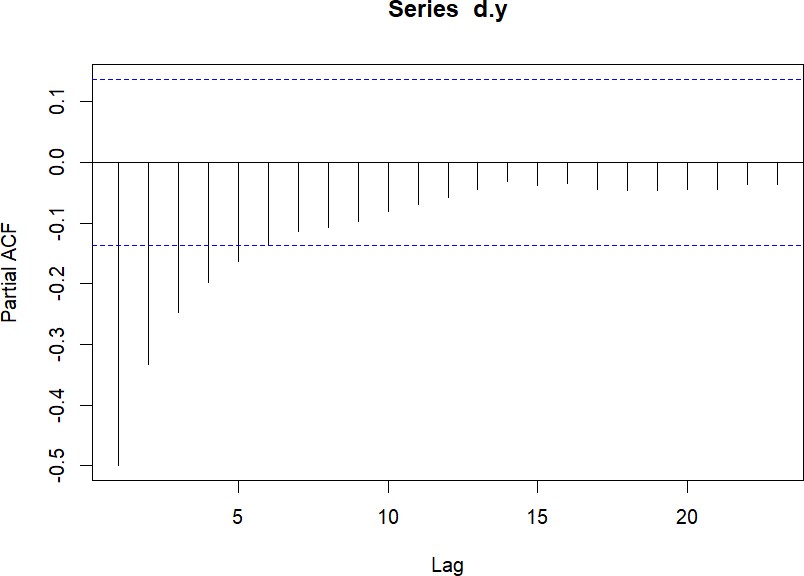
[71] 14598.77 14594.64 14590.69 14586.91 14583.29 14579.82 14576.50 14573.32 14570.28 14567.36

[81] 14564.57 14561.90 14559.34 14556.89 14554.55 14552.30 14550.15 14548.09 14546.12 14544.23 [91] 14542.42 14540.69 14539.04 14537.45 14535.93 14534.47 14533.08 14531.75 14530.47 14529.25 > plot(mydata.pred1$pred)









II . Analysis of weekly SENSEX indices of BSE :

1.MOVING AVERAGE CODE :

**# Create a data frame from the provided data**

**data.df <- data.frame(**

**week = 1:52,**

**year2019 = c(35850, 35853, 36578, 35656, 36582, 36395, 35498, 36213, 36063, 37054, 38095, 37808, 38871, 38700, 38905, 38645,**

**39067, 38600, 37090, 39352, 39683, 40267, 39784, 38960, 39122, 39686, 38720, 38896, 38031, 37686, 36699, 37581,**

**37402, 37494, 37332, 37145, 37123, 39090, 38667, 37531, 38214, 39298, 39058, 40301, 40345, 40284, 40889, 40802,**

**40487, 40938, 41642, 41558),**

**year2020 = c(41859, 41528, 41155, 39872, 40979, 41055, 40363, 38144, 35634, 31390, 25981, 28440, 27590, 30690, 31648, 31743,**

**31715, 31561, 30028, 30672, 33303, 34370, 33228, 34911, 34961, 36487, 36693, 37418, 37434, 36939, 38182, 38050,**

**38799, 38628, 38417, 38756, 38034, 37981, 38973, 40593, 40431, 40145, 39757, 42597, 43443, 44077, 44149, 45426,**

**46253, 45553, 47353, 47751),**

**year2021 = c(48093, 49584, 49624, 46874, 50614, 51531, 51324, 51039, 50846, 51279, 49216, 48440, 50029, 49746, 40083, 48080,**

**49765, 48949, 48690, 49564, 51115, 52232, 52300, 52323, 52699, 52318, 52568, 53158, 52837, 52653, 54492, 54843,**

**55629, 55949, 57852, 58305, 59141, 59885, 59126, 59677, 61305, 60923, 59984, 60067, 59919, 59636, 58795, 58461,**

**58807, 57901, 57315, 57794),**

**year2022 = c(59601, 61235, 59464, 57276, 58788, 58926, 57892, 54529, 55102, 55464, 57863, 57595, 58568, 59034, 58338, 57911,**

**57521, 55702, 52930, 52792, 54252, 58189, 55320, 51495, 52265, 53018, 54178, 53416, 55681, 56857, 58298, 59332,**

**60298, 58774, 58766, 59688, 59934, 59119, 56409, 58222, 57235, 59202, 59756, 60836, 60613, 61750, 62272, 63284,**

**62570, 61799, 60826, 61133)**

**)**

**# Convert the data into a time series**

**data.ts <- ts(data = unlist(data.df[, -1]), frequency = 52, start = c(2019, 1))**

**# Calculate the moving average forecast**

**sma <- forecast::ma(data.ts, 5)**

**myforecast <- forecast::forecast(sma, 20)**

**# Print the moving average forecast**

**print("Moving Average Forecast:")**

**print(myforecast)**

**# Plot the moving average forecast**

**plot(myforecast, main = "Moving Average Forecast")**

**OUTPUT :**

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

2023.429 37188.06 36923.36 37452.76 36783.23 37592.89

2023.571 36896.11 36343.43 37448.78 36050.86 37741.35

2023.714 36604.15 35705.63 37502.68 35229.97 37978.33

2023.857 36312.20 35018.62 37605.78 34333.84 38290.56

2024.000 36020.24 34288.34 37752.15 33371.52 38668.97

2024.143 35728.29 33519.12 37937.46 32349.66 39106.92

2024.286 35436.34 32714.34 38158.33 31273.41 39599.27

2024.429 35144.38 31876.69 38412.07 30146.88 40141.88

2024.571 34852.43 31008.41 38696.45 28973.50 40731.35

2024.714 34560.48 30111.37 39009.58 27756.16 41364.80

2024.857 34268.52 29187.20 39349.84 26497.31 42039.73

2025.000 33976.57 28237.30 39715.83 25199.12 42754.01

2025.143 33684.61 27262.92 40106.31 23863.48 43505.75

2025.286 33392.66 26265.14 40520.18 22492.06 44293.26

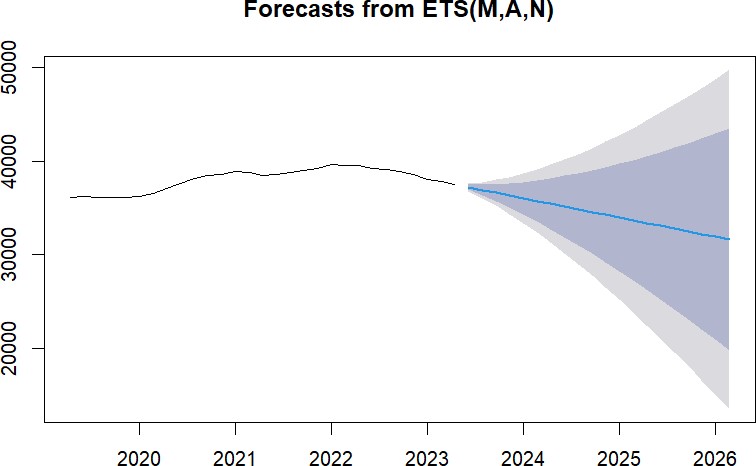
2025.429 33100.71 25244.96 40956.46 21086.37 45115.04

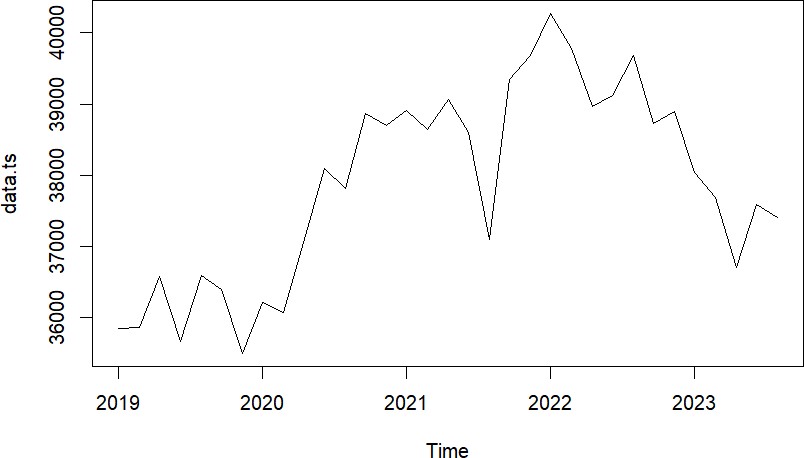
2025.571 32808.75 24203.24 41414.27 19647.75 45969.75

2025.714 32516.80 23140.78 41892.81 18177.42 46856.18

2025.857 32224.84 22058.31 42391.38 16676.47 47773.22

2026.000 31932.89 20956.47 42909.31 15145.91 48719.88 2026.143 31640.94 19835.86 43446.01 13586.64 49695.24





**2.Holt’s Single and double exponential CODE :**

**# Load necessary libraries**

**library(forecast)**

**library(ggplot2)**

**# Create a data frame from the provided data**

**data.df <- data.frame(**

**week = 1:52,**

**year2019 = c(35850, 35853, 36578, 35656, 36582, 36395, 35498, 36213, 36063, 37054, 38095, 37808, 38871, 38700, 38905, 38645,**

**39067, 38600, 37090, 39352, 39683, 40267, 39784, 38960, 39122, 39686, 38720, 38896, 38031, 37686, 36699, 37581,**

**37402, 37494, 37332, 37145, 37123, 39090, 38667, 37531, 38214, 39298, 39058, 40301, 40345, 40284, 40889, 40802,**

**40487, 40938, 41642, 41558),**

**year2020 = c(41859, 41528, 41155, 39872, 40979, 41055, 40363, 38144, 35634, 31390, 25981, 28440, 27590, 30690, 31648, 31743,**

**31715, 31561, 30028, 30672, 33303, 34370, 33228, 34911, 34961, 36487, 36693, 37418, 37434, 36939, 38182, 38050,**

**38799, 38628, 38417, 38756, 38034, 37981, 38973, 40593, 40431, 40145, 39757, 42597, 43443, 44077, 44149, 45426,**

**46253, 45553, 47353, 47751),**

**year2021 = c(48093, 49584, 49624, 46874, 50614, 51531, 51324, 51039, 50846, 51279, 49216, 48440, 50029, 49746, 40083, 48080,**

**49765, 48949, 48690, 49564, 51115, 52232, 52300, 52323, 52699, 52318, 52568, 53158, 52837, 52653, 54492, 54843,**

**55629, 55949, 57852, 58305, 59141, 59885, 59126, 59677, 61305, 60923, 59984, 60067, 59919, 59636, 58795, 58461,**

**58807, 57901, 57315, 57794),**

**year2022 = c(59601, 61235, 59464, 57276, 58788, 58926, 57892, 54529, 55102, 55464, 57863, 57595, 58568, 59034, 58338, 57911,**

**57521, 55702, 52930, 52792, 54252, 58189, 55320, 51495, 52265, 53018, 54178, 53416, 55681, 56857, 58298, 59332,**

**60298, 58774, 58766, 59688, 59934, 59119, 56409, 58222, 57235, 59202, 59756, 60836, 60613, 61750, 62272, 63284,**

**62570, 61799, 60826, 61133)**

**)**

**# Convert the data into a time series**

**data.ts <- ts(data = unlist(data.df[, -1]), frequency = 52, start = c(2019, 1))**

**# Single Exponential Smoothing Forecast**

**ses\_model <- forecast::ses(data.ts)**

**ses\_forecast <- forecast::forecast(ses\_model, h = 20)**

**# Double Exponential Smoothing (Holt's) Forecast**

**des\_model <- forecast::holt(data.ts, h = 20)**

**des\_forecast <- forecast::forecast(des\_model)**

**# Print forecasts**

**cat("Single Exponential Smoothing Forecast:\n")**

**print(ses\_forecast)**

**cat("\nDouble Exponential Smoothing (Holt's) Forecast:\n")**

**print(des\_forecast)**

**# Plot the forecasts**

**plot(ses\_forecast, main = "Single Exponential Smoothing Forecast")**

**plot(des\_forecast, main = "Double Exponential Smoothing (Holt's) Forecast")**

**OUTPUT :**

d.y=diff(y2)

> summary(y1)

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

25981 38509 46874 46903 57296 63284 1

> y2=na.omit(y1)

> summary(y2)

Min. 1st Qu. Median Mean 3rd Qu. Max.

25981 38509 46874 46903 57296 63284

> y3=ts(y2)

> library(forecast)

> plot(y3)

> #single exponential

> model1=forecast::ets(y3,'ANN')

> forc1=forecast(model1,8)

> forc1

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

1. 61115.93 59131.06 63100.79 58080.33 64151.52
2. 61115.93 58410.91 63820.94 56978.96 65252.89
3. 61115.93 57845.68 64386.17 56114.52 66117.33
4. 61115.93 57364.67 64867.18 55378.88 66852.97
5. 61115.93 56938.69 65293.16 54727.39 67504.46
6. 61115.93 56552.30 65679.55 54136.46 68095.39
7. 61115.93 56196.16 66035.69 53591.79 68640.06 215 61115.93 55864.12 66367.74 53083.98 69147.88

> plot(forc1)

> forecast::accuracy(model1)

ME RMSE MAE MPE MAPE MASE A

CF1

Training set 129.2703 1541.3 1004.263 0.1987584 2.257596 1.000838 0.001046

254

> #double exponential

> mod2=forecast::ets(y3,'AAN')

> forc2=forecast(mod2,8)

> plot(forc2)

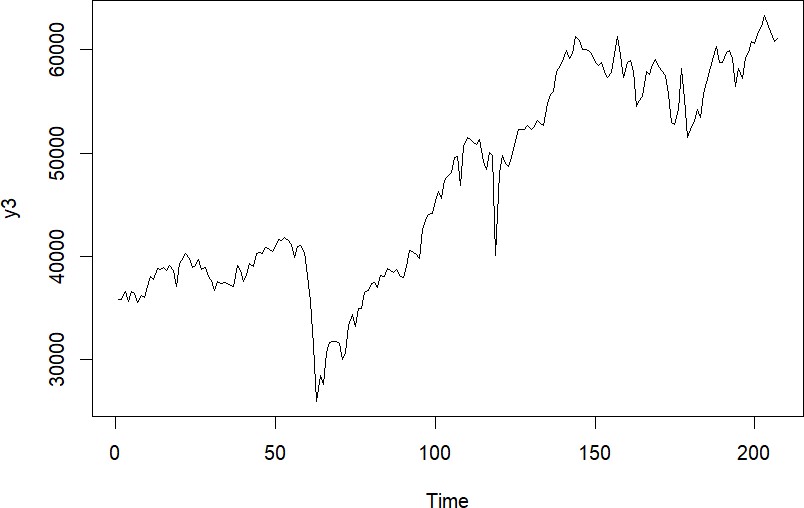
> forecast::accuracy(mod2)

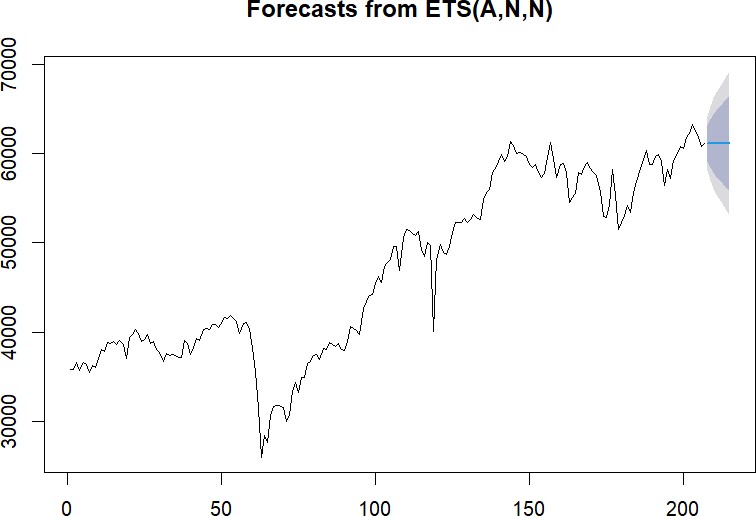
ME RMSE MAE MPE MAPE MASE

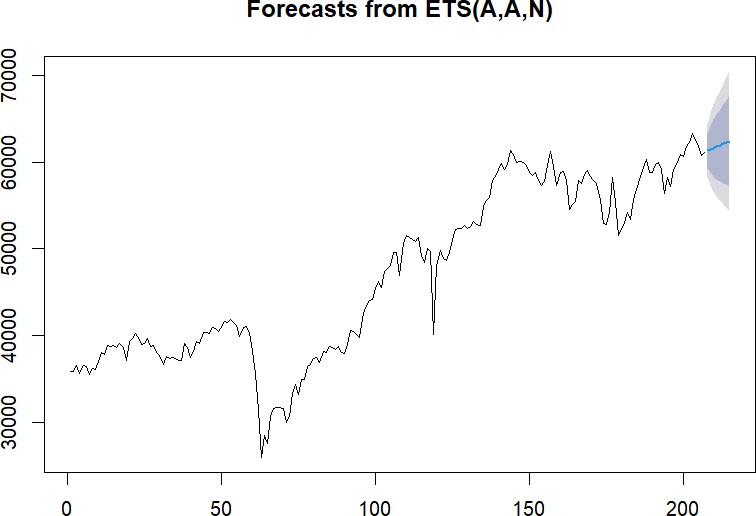
ACF1

Training set -45.19132 1535.968 996.7562 -0.1898095 2.248791 0.9933566 0.0

08492406







3.ARIMA

CODE :

**data.df=read.csv("C:/Users/LENOVO/OneDrive/Documents/pd.csv"**

**)**

**data.df library(tseries) attach(data.df) y=sales y1=as.numeric(y) d.y=diff(y2) summary(y1)**

**y2=na.omit(y1) summary(y2) plot(d.y)**

**adf.test(y2,alternative = "stationary",k=0) adf.test(y2,alternative = "explosive",k=0)**

**arima(y2,order = c(1,0,0))**

**adf.test(d.y,k=0) adf.test(d.y) acf(y2) pacf(y2)**

**acf(d.y) pacf(d.y)**

**arima(y2,order = c(1,0,1))**

**mydata.arima101=arima(y2,order = c(1,0,1)) mydata.pred1=predict(mydata.arima101,n.ahead=100) mydata.pred1$pred plot(mydata.pred1$pred)**

**output :**

library(tseries)

> attach(data.df)

The following objects are masked from data.df (pos = 7):

date, sales

The following objects are masked from data.df (pos = 10):

date, sales

The following objects are masked from data.df (pos = 11):

date, sales

The following objects are masked from data.df (pos = 13):

date, sales

The following objects are masked from data.df (pos = 14):

date, sales

The following objects are masked from data.df (pos = 15):

date, sales

> y=sales

> y1=as.numeric(y)

> d.y=diff(y2)

> summary(y1)

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

25981 38509 46874 46903 57296 63284 1

> y2=na.omit(y1)

> summary(y2)

Min. 1st Qu. Median Mean 3rd Qu. Max.

25981 38509 46874 46903 57296 63284

> plot(d.y)

> adf.test(y2,alternative = "stationary",k=0) Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -2.4831, Lag order = 0, p-value = 0.3736 alternative hypothesis: stationary

> adf.test(y2,alternative = "explosive",k=0) Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -2.4831, Lag order = 0, p-value = 0.6264 alternative hypothesis: explosive

>

> arima(y2,order = c(1,0,0)) Call:

arima(x = y2, order = c(1, 0, 0))

Coefficients: ar1 intercept 0.9896 47707.648

s.e. 0.0088 7465.748 sigma^2 estimated as 2386194: log likelihood = -1815.58, aic = 3637.16

> adf.test(d.y,k=0)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -24.585, Lag order = 0, p-value = 0.01 alternative hypothesis: stationary

> adf.test(d.y)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -9.6904, Lag order = 5, p-value = 0.01 alternative hypothesis: stationary

> acf(y2)

> pacf(y2)

> acf(d.y)

> pacf(d.y)

> arima(y2,order = c(1,0,1)) Call:

arima(x = y2, order = c(1, 0, 1))

Coefficients:

ar1 ma1 intercept

0.9914 -0.0666 48173.961

s.e. 0.0081 0.0786 8042.349 sigma^2 estimated as 2377514: log likelihood = -1815.23, aic = 3638.46

> mydata.arima101=arima(y2,order = c(1,0,1))

> mydata.pred1=predict(mydata.arima101,n.ahead=100)

> mydata.pred1$pred

Time Series:

Start = 208

End = 307

Frequency = 1

[1] 60997.48 60886.87 60777.21 60668.49 60560.71 60453.86 60347.93 60242 .92 60138.81 60035.60

[11] 59933.28 59831.84 59731.28 59631.59 59532.75 59434.77 59337.63 59241

.34 59145.87 59051.22

[21] 58957.39 58864.38 58772.16 58680.74 58590.11 58500.26 58411.18 58322

.87 58235.33 58148.54

[31] 58062.50 57977.20 57892.63 57808.80 57725.69 57643.29 57561.61 57480

.63 57400.35 57320.76

[41] 57241.86 57163.64 57086.10 57009.22 56933.01 56857.45 56782.54 56708

.29 56634.67 56561.69

[51] 56489.33 56417.60 56346.49 56276.00 56206.11 56136.82 56068.13 56000

.04 55932.53 55865.60

[61] 55799.25 55733.48 55668.27 55603.62 55539.53 55476.00 55413.01 55350

.56 55288.66 55227.29

[71] 55166.44 55106.12 55046.33 54987.05 54928.28 54870.01 54812.25 54754

.99 54698.22 54641.94

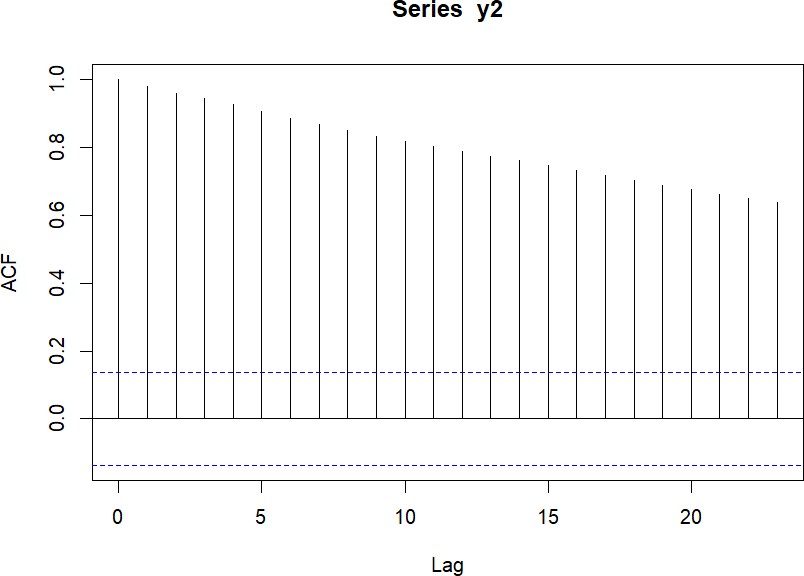
[81] 54586.15 54530.84 54476.00 54421.64 54367.75 54314.32 54261.35 54208

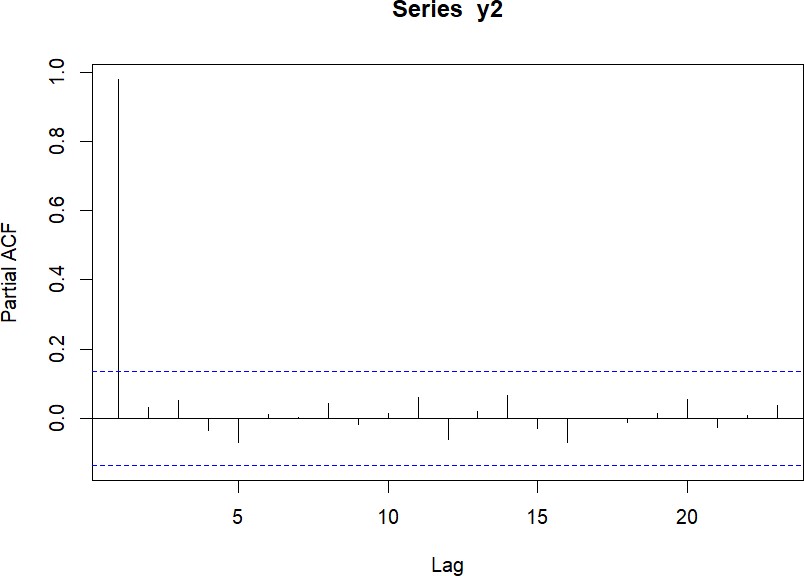
.84 54156.78 54105.17

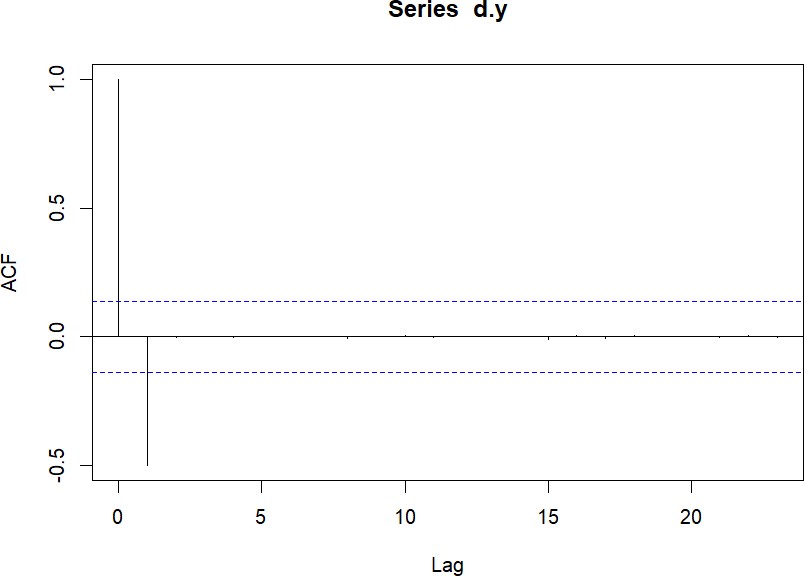
[91] 54054.01 54003.29 53953.01 53903.15 53853.73 53804.74 53756.17 53708

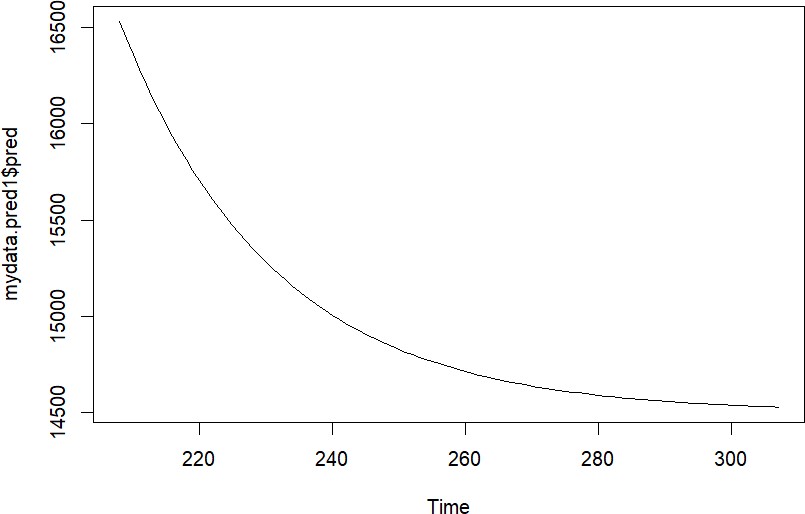
.02 53660.28 53612.95

> plot(mydata.pred1$pred)









**CHAPTER-IV**

**4.1 DATA COLLECTION:**

We have collected the monthly NIFTY indices for the period

2013 – 2022 and weekly NIFTY indices for the 2019 – 2022 on

NATIONAL STOCK EXCHANGE (NSE) from the website of NSE and are presented below in table 3 and table 4.

TABLE 3: MONTHLY NIFTY INDICES OF NSE FOR THE PERIOD JAN 2013 – DEC 2022

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Month | 2013 | 2014 | 2015 | 2016 | 2017 | 2018 | 2019 | 2020 | 2021 | 2022 |
| Jan | 6034 | 6089 | 8808 | 7963 | 8561 | 11027 | 10813 | 11962 | 13634 | 17339 |
| Feb | 5693 | 6276 | 8844 | 6987 | 8879 | 10492 | 10792 | 11201 | 14529 | 16793 |
| Mar | 5682 | 6704 | 8491 | 7738 | 9173 | 10113 | 11623 | 8597 | 14690 | 174564 |
| Apr | 5930 | 6696 | 8181 | 7849 | 9304 | 10739 | 11748 | 9859 | 14631 | 17102 |
| May | 5930 | 7229 | 8433 | 8160 | 9621 | 10736 | 11922 | 9580 | 15582 | 16584 |
| Jun | 5985 | 7611 | 8368 | 8287 | 9520 | 10714 | 11788 | 10302 | 15721 | 15708 |
| Jul | 5842 | 7721 | 8532 | 8638 | 10077 | 11356 | 11118 | 11073 | 15763 | 17158 |
| Aug | 5742 | 7954 | 7971 | 8786 | 9917 | 11680 | 11023 | 11387 | 17132 | 17759 |
| Sep | 5471 | 7964 | 7948 | 8611 | 9788 | 10930 | 11474 | 11847 | 17618 | 17094 |
| Oct | 5735 | 8322 | 8065 | 8638 | 10335 | 10386 | 11877 | 11642 | 17671 | 18012 |
| Nov | 6299 | 8588 | 7935 | 8224 | 10226 | 10872 | 12056 | 12698 | 16983 | 18718 |
| Dec | 6176 | 8282 | 7946 | 8185 | 10530 | 10862 | 12168 | 13981 | 17354 | 18105 |

TABLE 4 : WEEKLY NIFTY INDICES OF NSE FOR THE YEARS 2019 – 2022

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 2019 | 2020 | 2021 | 2022 |  | 2019 | 2020 | 2021 | 2022 |
| 1 | 10771 | 11993 | 14137 | 17745 | 27 | 11558 | 10763 | 15727 | 16132 |
| 2 | 10737 | 12329 | 14595 | 18257 | 28 | 11588 | 10802 | 15924 | 15938 |
| 3 | 10961 | 12224 | 14590 | 17757 | 29 | 11346 | 11022 | 15824 | 16605 |
| 4 | 10661 | 12019 | 13817 | 17110 | 30 | 11189 | 11131 | 15778 | 16929 |
| 5 | 10912 | 11707 | 14895 | 17560 | 31 | 10862 | 10891 | 16294 | 17382 |
| 6 | 10888 | 12031 | 15173 | 17605 | 32 | 11109 | 11270 | 16364 | 17659 |
| 7 | 10640 | 12045 | 15118 | 17304 | 33 | 11053 | 11247 | 16568 | 17956 |
| 8 | 10880 | 11829 | 15097 | 16247 | 34 | 11057 | 11466 | 16636 | 17522 |
| 9 | 10863 | 11132 | 15080 | 16498 | 35 | 11023 | 11387 | 17234 | 17542 |
| 10 | 11168 | 10451 | 15174 | 16594 | 36 | 11003 | 11355 | 17369 | 17798 |
| 11 | 11462 | 9197 | 14557 | 17287 | 37 | 11003 | 11440 | 17629 | 17877 |
| 12 | 11354 | 7610 | 14324 | 17222 | 38 | 11600 | 11250 | 17822 | 17629 |
| 13 | 11669 | 8281 | 14867 | 17464 | 39 | 11474 | 11227 | 17618 | 16818 |
| 14 | 11604 | 8083 | 14873 | 17639 | 40 | 11126 | 11503 | 17790 | 17331 |
| 15 | 11690 | 8993 | 14581 | 17475 | 41 | 11341 | 11930 | 18338 | 17014 |
| 16 | 11594 | 9261 | 14406 | 17392 | 42 | 11661 | 11873 | 18178 | 17563 |
| 17 | 11754 | 9282 | 14894 | 17245 | 43 | 11583 | 11767 | 17857 | 17736 |
| 18 | 11598 | 9293 | 14724 | 16682 | 44 | 11941 | 11669 | 17916 | 18052 |
| 19 | 11148 | 9239 | 14696 | 15808 | 45 | 11913 | 12461 | 17873 | 18028 |
| 20 | 11828 | 8823 | 14906 | 15809 | 46 | 11884 | 12719 | 17764 | 18343 |
| 21 | 11924 | 9039 | 15337 | 16170 | 47 | 12073 | 12926 | 17536 | 18484 |
| 22 | 12088 | 9826 | 15690 | 16628 | 48 | 123048 | 12968 | 17401 | 18812 |
| 23 | 11922 | 10167 | 15737 | 16478 | 49 | 11937 | 13355 | 17516 | 18609 |
| 24 | 11672 | 9813 | 15691 | 15360 | 50 | 12053 | 13558 | 17248 | 18414 |
| 25 | 11699 | 10311 | 15790 | 15556 | 51 | 12262 | 13328 | 17072 | 18127 |
| 26 | 11865 | 10312 | 15680 | 15718 | 52 | 12255 | 13873 | 17203 | 18191 |

4.2. DATA ANALYSIS :

In time series analysis there are many methods for handling, but here we adopt the following two methods for analysis and forecasting of the SENSEX indices and NIFTY indices.

1. moving average
2. HOLT’S single and double exponential method.
3. ARIMA method.

These methods are already explained elaborately in Chapter

– II. Further, the for selection of these methods is also explained in Chapter – I.

In this section, we carry out the time series analysis for the following stock market index data:

* Monthly NIFTY indices(table 3)
* Weekly NIFTY indices (table 4)

The sequence of various steps,those we have followed for the analysis of each of the above indices data,is as follows:

1. Plotting of time series graph
2. Analysis using moving average , single and double exponential method.
3. Modelling using of ARIMA method.
4. Comparison of single and double exponential and ARIMA models for selection of better

IN THE NEXT SECTION ,using the better model, we compute the forecasted monthly indices for the years 2013 – 2022 and forecasted weekly indices for the period 2019 – 2022.

**As** a first step , using R,we draw the time series plot for the BSE monthly SENSEX indices for the period **JAN 2019 – DEC 2022** and presented below.

**S**econd step enter the data in the excel and then write the R code.

**WE CARRY OUT THE ABOVE STEPS OF THE TIME SERIES ANALYSIS USING R**

**III . Analysis of monthly NIFTY indices of NSE:**

1.MOVING AVERAGE CODE:

data.df=read.csv("C:/Users/LENOVO/OneDrive/

Documents/pd.csv",header=TRUE,stringsAs

Factors = FALSE) class(data.df) colnames(data.df) class(data.df$Date) head(data.df)

data.ts=ts(data=data.df$Sales,frequency = 12, start=c(2013,01),end=c(2022,12)) class(data.ts) data.ts

sma=forecast::ma(data.ts,5) sma

myforecast=forecast::forecast(sma,20) myforecast plot(myforecast) plot(data.ts)

**OUTPUT :**

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

Nov 2022 17526.57 13980.6600 21072.47 12103.5701 22949.56

Dec 2022 17199.49 12390.2334 22008.75 9844.3661 24554.62

Jan 2023 23772.78 15666.8808 31878.67 11375.8792 36169.67

Feb 2023 22093.95 13390.3999 30797.51 8783.0170 35404.89

Mar 2023 19619.63 10956.5978 28282.67 6370.6647 32868.60

Apr 2023 17990.87 9257.8224 26723.91 4634.8280 31346.91

May 2023 18072.98 8557.7603 27588.21 3520.7056 32625.26

Jun 2023 17787.31 7730.4373 27844.18 2406.6507 33167.97

Jul 2023 17786.69 7068.9015 28504.48 1395.2473 34178.13

Aug 2023 17876.16 6465.0915 29287.24 424.4344 35327.89

Sep 2023 18108.42 5922.6032 30294.24 -528.1807 36745.03

Oct 2023 17914.68 5257.0604 30572.30 -1443.4799 37272.84

Nov 2023 17526.79 4568.0822 30485.49 -2291.8423 37345.42

Dec 2023 17199.71 3931.3015 30468.11 -3092.5684 37491.98

Jan 2024 23773.08 4686.0136 42860.14 -5418.0665 52964.22

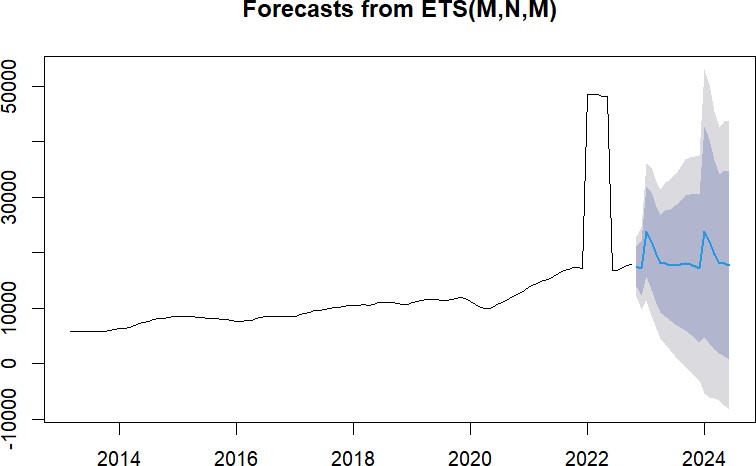
Feb 2024 22094.23 3671.8382 40516.62 -6080.3877 50268.85

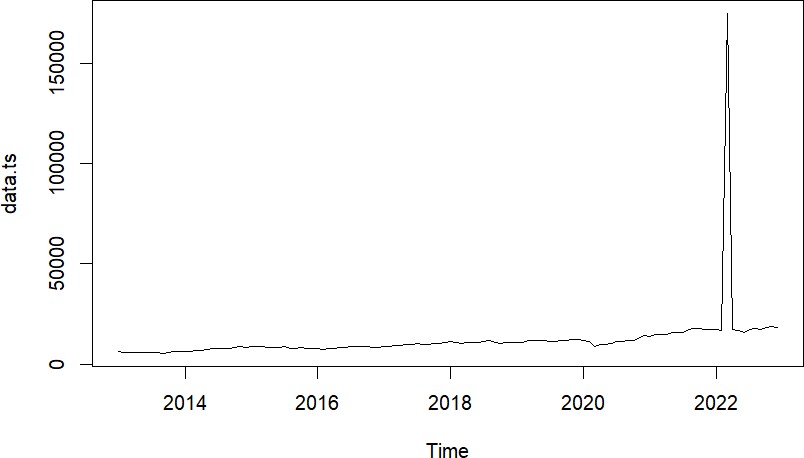
Mar 2024 19619.88 2662.8854 36576.87 -6313.6048 45553.36

Apr 2024 17991.09 1900.8567 34081.33 -6616.7999 42598.99

May 2024 18073.21 1372.3376 34774.08 -7468.5706 43614.99

Jun 2024 17787.53 827.2646 34747.80 -8150.9594 43726.03





### 2.Holt’s Single and double exponential

**CODE :**

**data.df=read.csv("C:/Users/LENOVO/OneDrive/Docume nts/pd.csv") data.df library(tseries) attach(data.df) y=sales y1=as.numeric(y) d.y=diff(y2) summary(y1) y2=na.omit(y1) summary(y2) y3=ts(y2)**

**library(forecast) plot(y3) #single exponential model1=forecast::ets(y3,'ANN') forc1=forecast(model1,8) forc1 plot(forc1) forecast::accuracy(model1) #double exponential mod2=forecast::ets(y3,'AAN') forc2=forecast(mod2,8) plot(forc2) forecast::accuracy(mod2)**

**> model1=forecast::ets(y3,'ANN')**

**> forc1=forecast(model1,8)**

**> forc1**

**OUTPUT :**

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

1. 21763.47 2639.484 40887.45 -7484.140 51011.07
2. 21763.47 2593.065 40933.87 -7555.133 51082.07
3. 21763.47 2546.758 40980.18 -7625.954 51152.89
4. 21763.47 2500.561 41026.37 -7696.605 51223.54
5. 21763.47 2454.476 41072.46 -7767.086 51294.02
6. 21763.47 2408.500 41118.43 -7837.400 51364.33 127 21763.47 2362.633 41164.30 -7907.548 51434.48 128 21763.47 2316.875 41210.06 -7977.530 51504.46

> plot(forc1)

> forecast::accuracy(model1)

ME RMSE MAE MPE MAPE MASE ACF1

Training set 1799.249 14797.65 2952.2 5.484076 14.10776 0.9757226 -0.0303388

> mod2=forecast::ets(y3,'AAN')

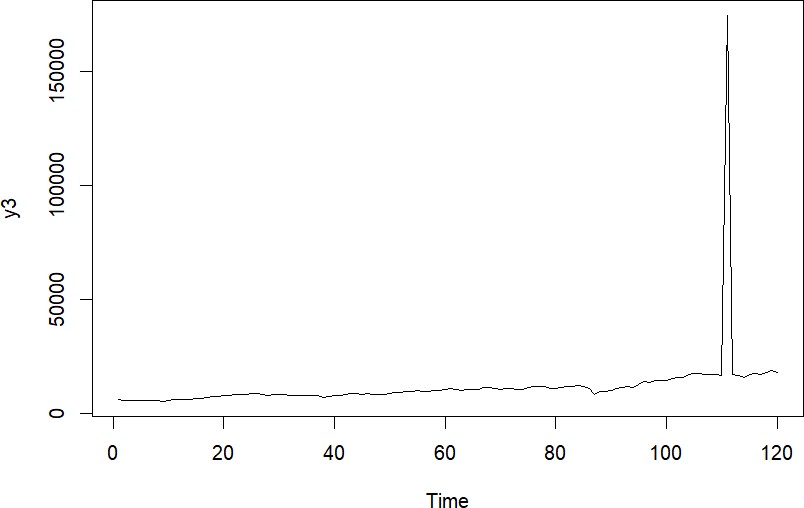
> forc2=forecast(mod2,8)

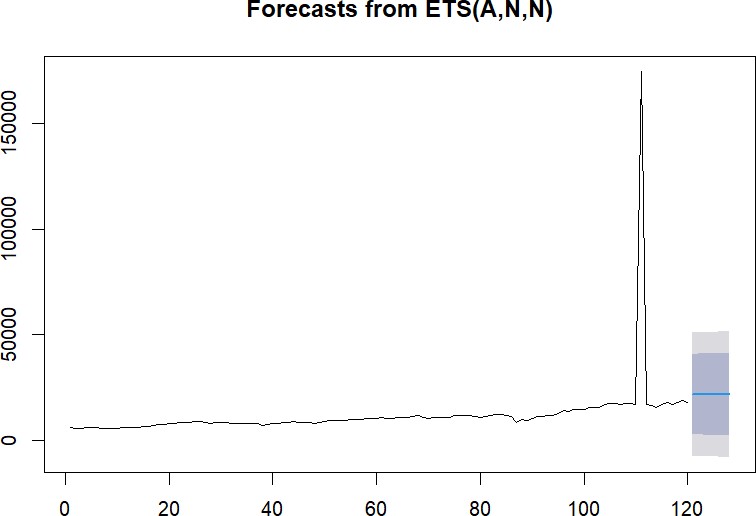
> plot(forc2)

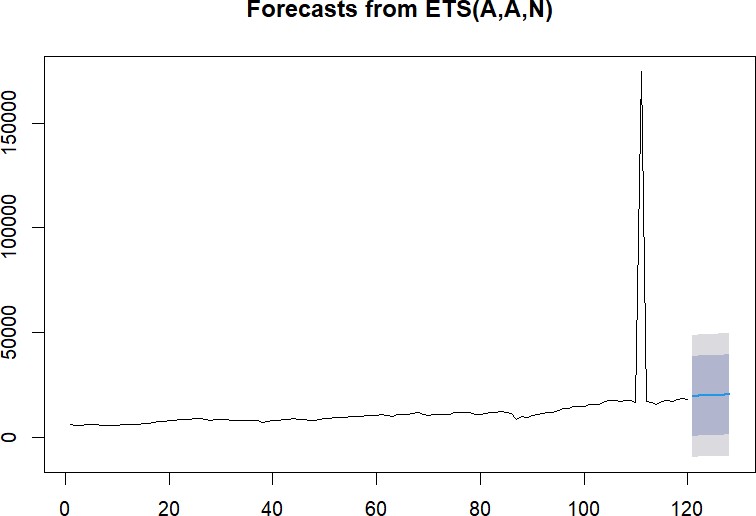
> forecast::accuracy(mod2)

ME RMSE MAE MPE MAPE MASE ACF1

Training set 1049.233 14629.9 2289.498 -3.27097 9.575577 0.7566948 0.0009500816







### 3.ARIMA

**CODE : data.df=read.csv("C:/Users/LENOVO/OneDrive/Documents/p d.csv") data.df library(tseries) attach(data.df) y=sales y1=as.numeric(y) d.y=diff(y2) summary(y1) y2=na.omit(y1) summary(y2) plot(d.y)**

**adf.test(y2,alternative = "stationary",k=0) adf.test(y2,alternative = "explosive",k=0) arima(y2,order = c(1,0,0)) adf.test(d.y,k=0) adf.test(d.y) acf(y2) pacf(y2) acf(d.y) pacf(d.y)**

**arima(y2,order = c(1,0,1))**

**mydata.arima101=arima(y2,order = c(1,0,1)) mydata.pred1=predict(mydata.arima101,n.ahead=100) mydata.pred1$pred plot(mydata.pred1$pred) OUTPUT :**

library(tseries)

> attach(data.df)

The following object is masked from data.df (pos = 4):

sales

The following object is masked from data.df (pos = 5):

sales

The following object is masked from data.df (pos = 10):

sales

The following object is masked from data.df (pos = 13):

sales

The following object is masked from data.df (pos = 14):

sales

The following object is masked from data.df (pos = 16):

sales

The following object is masked from data.df (pos = 17):

sales

The following object is masked from data.df (pos = 18):

sales

> y=sales

> y1=as.numeric(y)

> d.y=diff(y2)

> summary(y1)

Min. 1st Qu. Median Mean 3rd Qu. Max.

5471 8136 10095 11935 11888 174564

> y2=na.omit(y1)

> summary(y2)

Min. 1st Qu. Median Mean 3rd Qu. Max.

5471 8136 10095 11935 11888 174564

> plot(d.y)

> adf.test(y2,alternative = "stationary",k=0)

Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -10.77, Lag order = 0, p-value = 0.01 alternative hypothesis: stationary

> adf.test(y2,alternative = "explosive",k=0) Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -10.77, Lag order = 0, p-value = 0.99 alternative hypothesis: explosive

> arima(y2,order = c(1,0,0)) Call:

arima(x = y2, order = c(1, 0, 0))

Coefficients: ar1 intercept 0.1142 11934.922

s.e. 0.0904 1565.581 sigma^2 estimated as 231287890: log likelihood = -1325.83, aic = 2657.66

> adf.test(d.y,k=0)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -24.585, Lag order = 0, p-value = 0.01

alternative hypothesis: stationary

> adf.test(d.y)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -9.6904, Lag order = 5, p-value = 0.01 alternative hypothesis: stationary

> acf(y2)

> pacf(y2)

> acf(d.y)

> pacf(d.y)

> arima(y2,order = c(1,0,1)) Call:

arima(x = y2, order = c(1, 0, 1))

Coefficients:

ar1 ma1 intercept

0.9731 -0.9022 12758.564

s.e. 0.0420 0.0677 4208.446 sigma^2 estimated as 219305949: log likelihood = -1322.84, aic = 2653.67

> mydata.arima101=arima(y2,order = c(1,0,1))

> mydata.pred1=predict(mydata.arima101,n.ahead=100)

> mydata.pred1$pred

Time Series:

Start = 121

End = 220

Frequency = 1

[1] 20059.18 19862.97 19672.04 19486.24 19305.44 19129.49 18958.27 18791

.65 18629.51 18471.73

[11] 18318.19 18168.77 18023.37 17881.88 17744.19 17610.20 17479.82 17352

.93 17229.46 17109.30

[21] 16992.38 16878.59 16767.87 16660.12 16555.26 16453.23 16353.93 16257

.31 16163.28 16071.77

[31] 15982.73 15896.08 15811.76 15729.71 15649.86 15572.15 15496.54 15422

.96 15351.35 15281.67

[41] 15213.86 15147.87 15083.66 15021.17 14960.37 14901.19 14843.61 14787

.57 14733.04 14679.98

[51] 14628.34 14578.09 14529.19 14481.61 14435.30 14390.24 14346.39 14303

.71 14262.19 14221.78

[61] 14182.45 14144.19 14106.95 14070.71 14035.45 14001.13 13967.73 13935

.24 13903.62 13872.84

[71] 13842.90 13813.75 13785.40 13757.80 13730.95 13704.81 13679.38 13654

.63 13630.55 13607.12

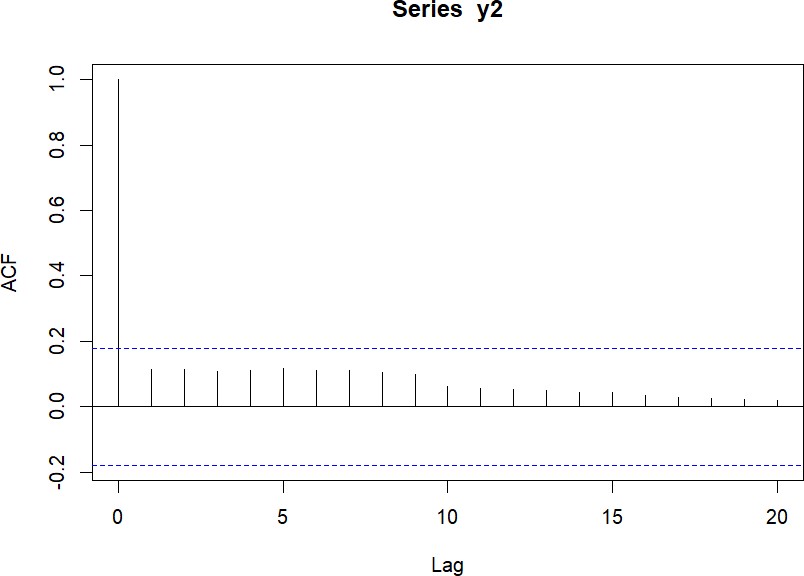
[81] 13584.31 13562.12 13540.53 13519.51 13499.06 13479.16 13459.79 13440

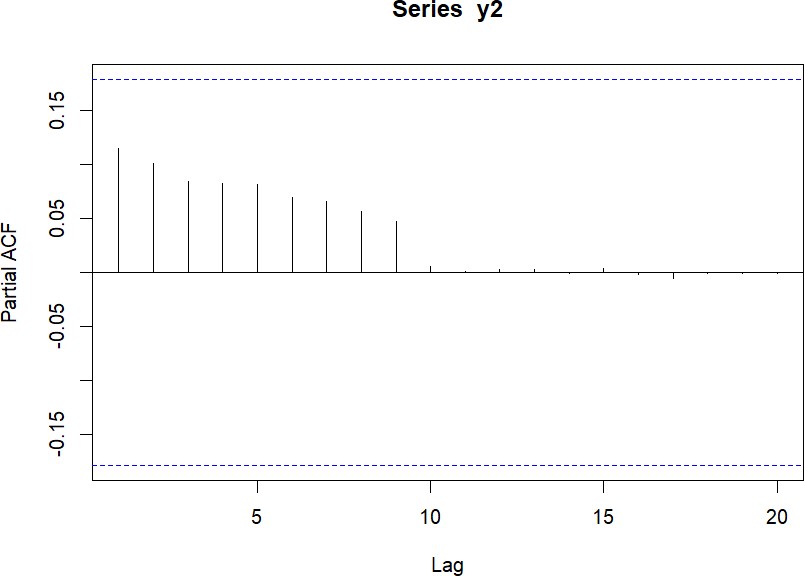
.95 13422.61 13404.76

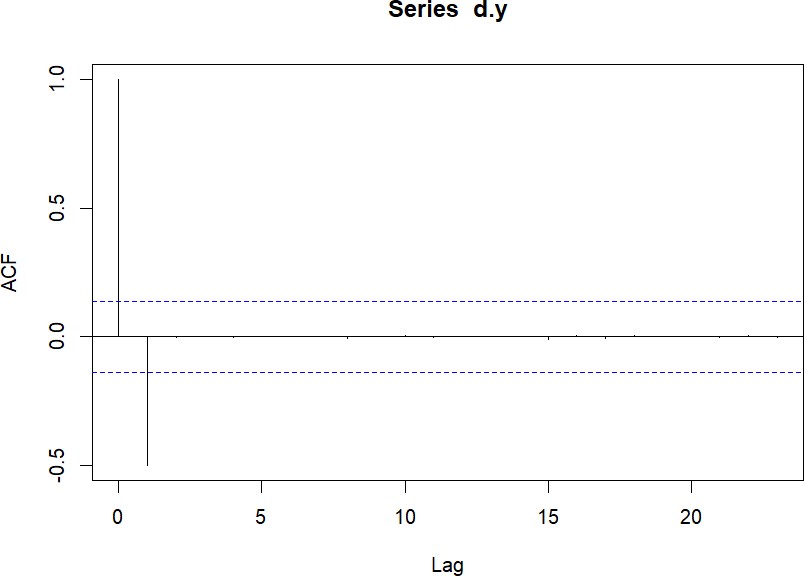
[91] 13387.40 13370.50 13354.05 13338.05 13322.47 13307.32 13292.57 13278

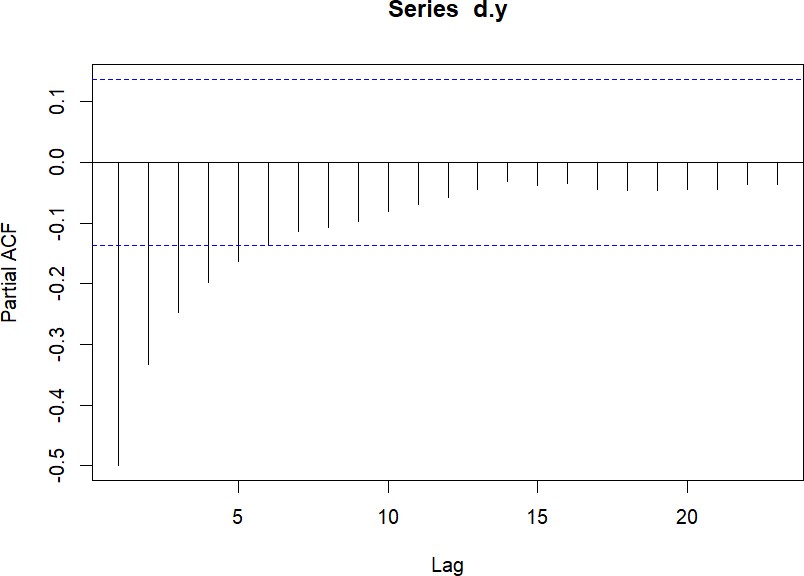
.22 13264.25 13250.66

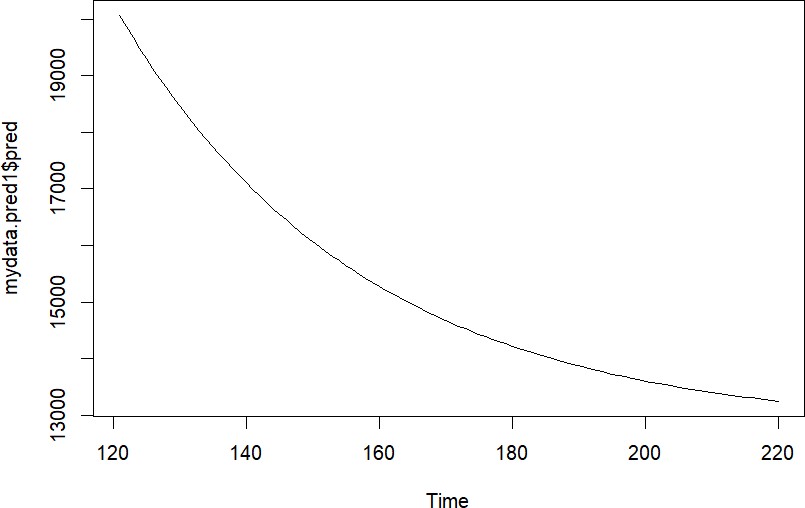
> plot(mydata.pred1$pred)











IV.Analysis of weekly NIFTY indices of NSE :

1.MOVING AVERAGE CODE:

data.df=read.csv("C:/Users/LENOVO/OneDrive/Doc uments/pd.csv",header=TRUE,stringsAsFactors = FALSE) class(data.df) colnames(data.df) class(data.df$Date) head(data.df)

data.ts=ts(data=data.df$Sales,frequency = 12,s tart=c(2013,01),end=c(2022,12)) class(data.ts) data.ts

sma=forecast::ma(data.ts,5) sma

myforecast=forecast::forecast(sma,20) myforecast plot(myforecast)

plot(data.ts)

**OUTPUT :**

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

2022.231 16822.78 14323.463 19322.10 13000.4050 20645.15

2022.250 16799.30 13264.918 20333.68 11393.9302 22204.67

2022.269 16952.68 12624.045 21281.32 10332.6019 23572.76

2022.288 16982.11 11983.872 21980.35 9337.9651 24626.25

2022.308 16995.16 11406.987 22583.33 8448.7890 25541.52

2022.327 17003.58 10882.071 23125.09 7641.5386 26365.63

2022.346 17062.45 10450.480 23674.42 6950.3138 27174.59

2022.365 17163.50 10095.016 24231.98 6353.1881 27973.81

2022.385 17301.60 9804.358 24798.85 5835.5561 28767.65

2022.404 17432.43 9529.649 25335.21 5346.1690 29518.69

2022.423 17577.66 9289.156 25866.15 4901.4890 30253.82

2022.442 17674.92 9017.867 26331.96 4435.1020 30914.73

2022.462 17698.69 8688.150 26709.22 3918.2600 31479.11

2022.481 17724.34 8373.675 27075.01 3423.7295 32024.96

2022.500 17786.07 8107.213 27464.93 2983.5352 32588.61

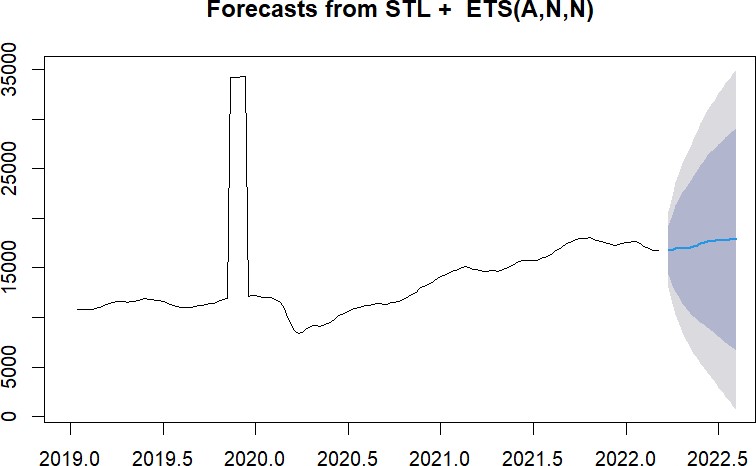
2022.519 17801.14 7804.860 27797.41 2513.1512 33089.12

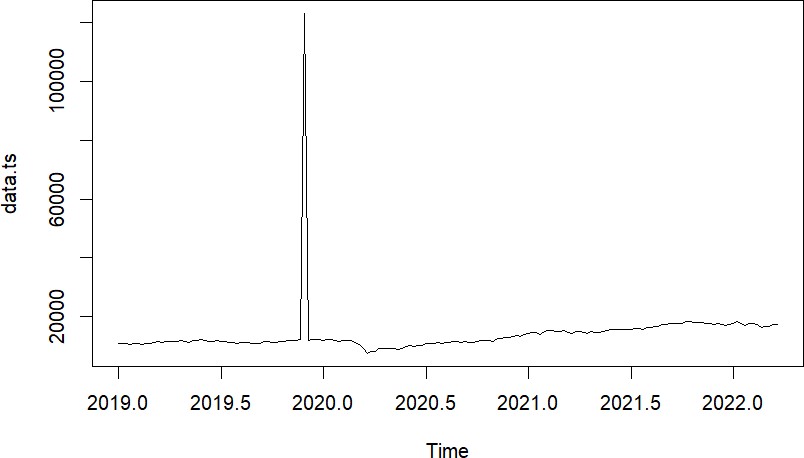
2022.538 17812.60 7508.677 28116.52 2054.1099 33571.09

2022.558 17827.89 7225.243 28430.53 1612.5415 34043.23

2022.577 17834.08 6940.902 28727.26 1174.4010 34493.76

2022.596 17870.16 6693.994 29046.32 777.6913 34962.62





### 2.Holt’s Single and double exponential

**CODE : data.df=read.csv("C:/Users/LENOVO/OneDrive/Docume nts/pd.csv") data.df library(tseries) attach(data.df) y=sales y1=as.numeric(y) d.y=diff(y2) summary(y1) y2=na.omit(y1) summary(y2) y3=ts(y2)**

**library(forecast) plot(y3)**

**#single exponential model1=forecast::ets(y3,'ANN') forc1=forecast(model1,8) forc1 plot(forc1) forecast::accuracy(model1)**

**#double exponential**

**mod2=forecast::ets(y3,'AAN') forc2=forecast(mod2,8) plot(forc2) forecast::accuracy(mod2) OUTPUT :**

class(y$sales)

[1] "integer"

> y1=as.numeric(y$sales)

> d.y=diff(y2)

> summary(y1)

Min. 1st Qu. Median Mean 3rd Qu. Max.

7610 11414 14137 14495 17091 123048

> y2=na.omit(y1)

> summary(y2)

Min. 1st Qu. Median Mean 3rd Qu. Max.

7610 11414 14137 14495 17091 123048

> y3=ts(y2)

> library(forecast)

> plot(y3)

> model1=forecast::ets(y3,'ANN')

> forc1=forecast(model1,8)

> forc1

Point Forecast Lo 80 Hi 80 Lo 95 Hi 95

1. 17416.21 7161.268 27671.15 1732.630 33099.79
2. 17416.21 7151.641 27680.78 1717.906 33114.51
3. 17416.21 7142.023 27690.40 1703.197 33129.22
4. 17416.21 7132.413 27700.00 1688.501 33143.92
5. 17416.21 7122.813 27709.60 1673.819 33158.60 213 17416.21 7113.222 27719.20 1659.150 33173.27

214 17416.21 7103.640 27728.78 1644.496 33187.92 215 17416.21 7094.067 27738.35 1629.854 33202.56

> plot(forc1)

> forecast::accuracy(model1)

ME RMSE MAE MPE MAPE MASE ACF1

Training set 618.8726 7963.223 1898.044 -1.232751 11.1471 1.410248 0.01187436

> mod2=forecast::ets(y3,'AAN')

> forc2=forecast(mod2,8)

> plot(forc2)

> forecast::accuracy(mod2)

ME RMSE MAE MPE MAPE MASE ACF1

Training set -119.5758 7862.034 1676.7 -6.380995 10.17515 1.24579 0.03480663

### 3.ARIMA

**CODE :**

**data.df=read.csv("C:/Users/LENOVO/OneDrive/Documents/pd.csv"**

**)**

**data.df library(tseries) attach(data.df) y=sales y1=as.numeric(y) d.y=diff(y2) summary(y1) y2=na.omit(y1) summary(y2) plot(d.y)**

**adf.test(y2,alternative = "stationary",k=0) adf.test(y2,alternative = "explosive",k=0) arima(y2,order = c(1,0,0)) adf.test(d.y,k=0) adf.test(d.y) acf(y2) pacf(y2) acf(d.y) pacf(d.y)**

**arima(y2,order = c(1,0,1))**

**mydata.arima101=arima(y2,order = c(1,0,1)) mydata.pred1=predict(mydata.arima101,n.ahead=100) mydata.pred1$pred plot(mydata.pred1$pred) OUTPUT :**

library(tseries)

> attach(data.df)

The following object is masked from data.df (pos = 3):

sales

The following object is masked from data.df (pos = 8):

sales

The following object is masked from data.df (pos = 11):

sales

The following object is masked from data.df (pos = 12):

sales

The following object is masked from data.df (pos = 14):

sales

The following object is masked from data.df (pos = 15):

sales

The following object is masked from data.df (pos = 16):

sales

> y=sales

> y1=as.numeric(y)

> d.y=diff(y2)

> summary(y1)

Min. 1st Qu. Median Mean 3rd Qu. Max.

7610 11414 14137 14495 17091 123048

> y2=na.omit(y1)

> summary(y2)

Min. 1st Qu. Median Mean 3rd Qu. Max.

7610 11414 14137 14495 17091 123048

> plot(d.y)

> adf.test(y2,alternative = "stationary",k=0) Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -13.814, Lag order = 0, p-value = 0.01 alternative hypothesis: stationary

> adf.test(y2,alternative = "explosive",k=0) Augmented Dickey-Fuller Test data: y2

Dickey-Fuller = -13.814, Lag order = 0, p-value = 0.99 alternative hypothesis: explosive

> arima(y2,order = c(1,0,0)) Call:

arima(x = y2, order = c(1, 0, 0))

Coefficients: ar1 intercept 0.0937 14495.2462

s.e. 0.0691 619.4745 sigma^2 estimated as 65310682: log likelihood = -2156.17, aic = 4318.35

>

> adf.test(d.y,k=0)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -15.161, Lag order = 0, p-value = 0.01 alternative hypothesis: stationary

> adf.test(d.y)

Augmented Dickey-Fuller Test

data: d.y

Dickey-Fuller = -5.8356, Lag order = 5, p-value = 0.01 alternative hypothesis: stationary

> acf(y2)

> pacf(y2)

> acf(d.y)

> pacf(d.y)

> arima(y2,order = c(1,0,1)) Call:

arima(x = y2, order = c(1, 0, 1))

Coefficients:

ar1 ma1 intercept

0.9575 -0.8961 14501.686

s.e. 0.0561 0.0860 1267.794

sigma^2 estimated as 63168855: log likelihood = -2152.82, aic = 4313.64

> mydata.arima101=arima(y2,order = c(1,0,1))

> mydata.pred1=predict(mydata.arima101,n.ahead=100)

> mydata.pred1$pred

Time Series:

Start = 208

End = 307

Frequency = 1

[1] 16530.05 16443.86 16361.33 16282.31 16206.65 16134.21 16064.84 15998

.42 15934.82 15873.93

[11] 15815.62 15759.79 15706.33 15655.14 15606.13 15559.20 15514.27 15471

.24 15430.04 15390.60

[21] 15352.83 15316.66 15282.03 15248.87 15217.13 15186.73 15157.62 15129

.75 15103.06 15077.51

[31] 15053.04 15029.61 15007.18 14985.70 14965.13 14945.44 14926.59 14908

.53 14891.24 14874.69

[41] 14858.84 14843.67 14829.13 14815.22 14801.90 14789.14 14776.93 14765

.23 14754.03 14743.31

[51] 14733.04 14723.21 14713.80 14704.79 14696.16 14687.89 14679.98 14672

.41 14665.15 14658.21

[61] 14651.56 14645.19 14639.09 14633.25 14627.66 14622.31 14617.18 14612

.28 14607.58 14603.08

[71] 14598.77 14594.64 14590.69 14586.91 14583.29 14579.82 14576.50 14573

.32 14570.28 14567.36

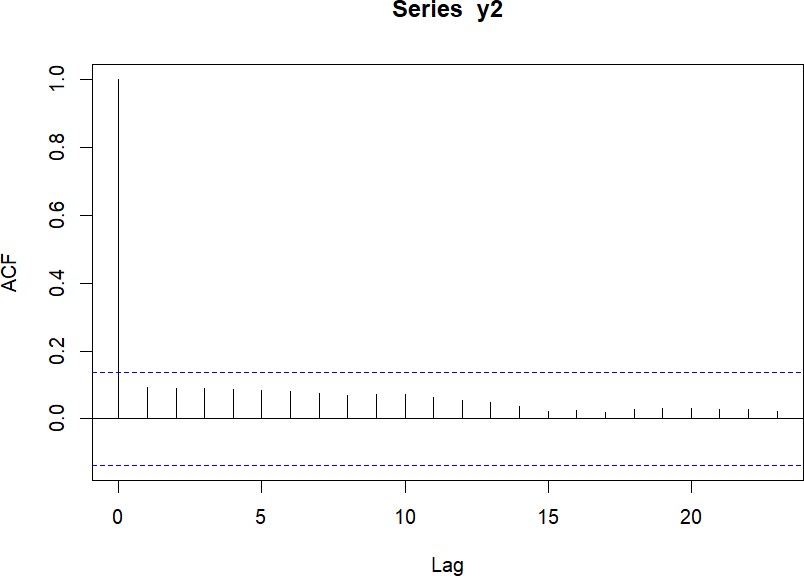
[81] 14564.57 14561.90 14559.34 14556.89 14554.55 14552.30 14550.15 14548

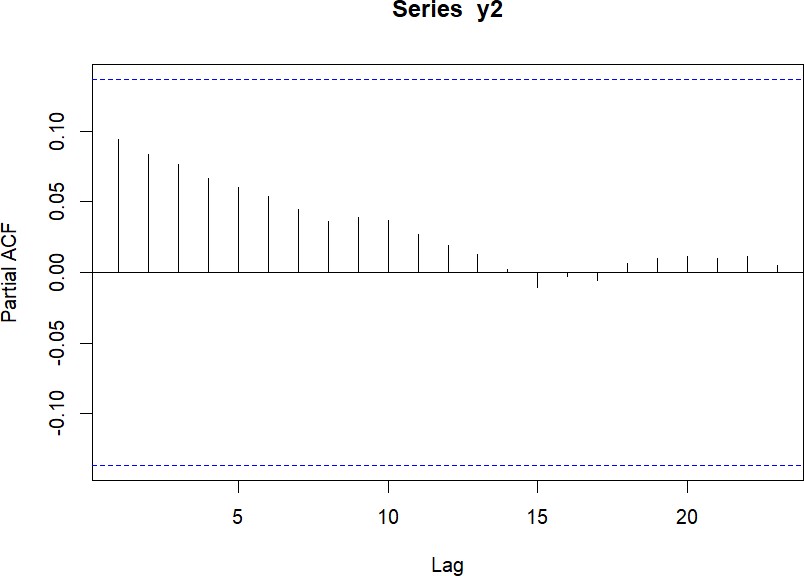
.09 14546.12 14544.23

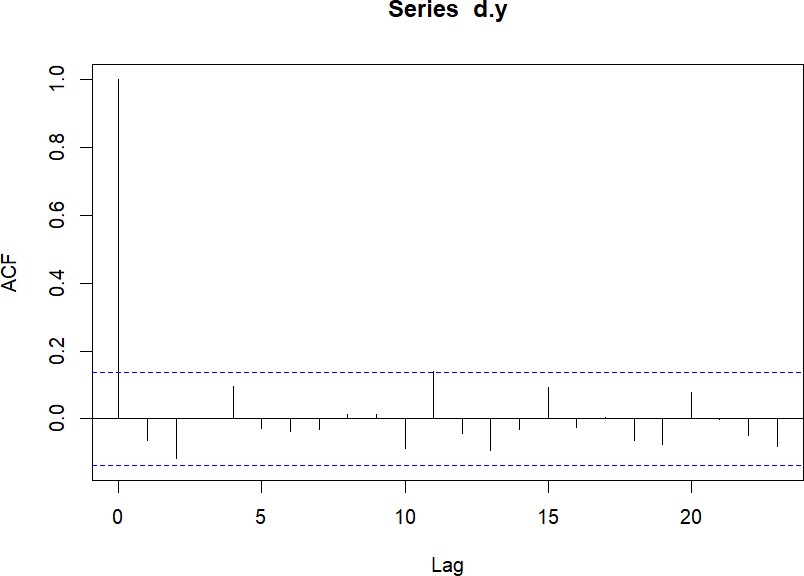
[91] 14542.42 14540.69 14539.04 14537.45 14535.93 14534.47 14533.08 14531

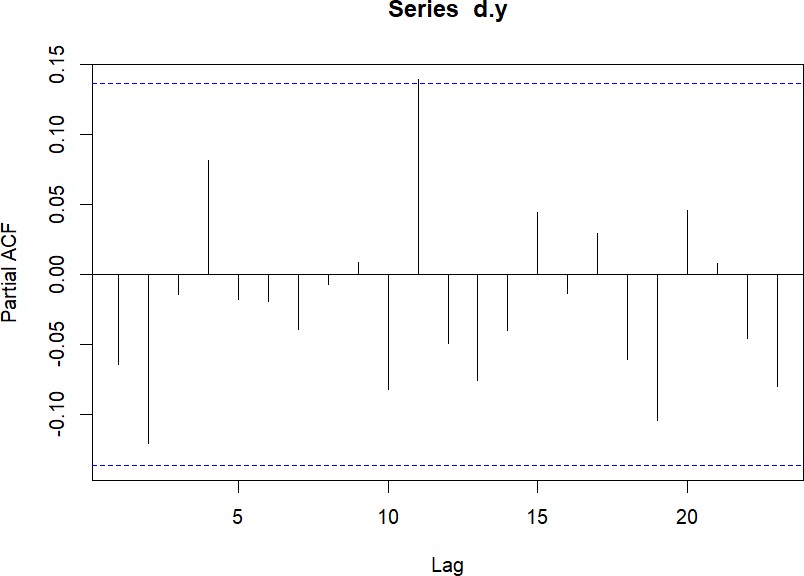
.75 14530.47 14529.25

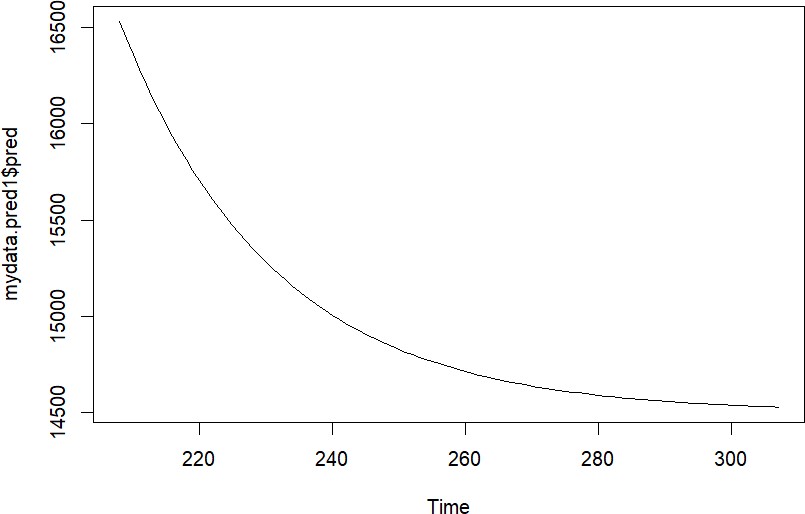
> plot(mydata.pred1$pred)











**CHAPTER-V**

**OBSERVATION and CONCLUSION:**

Both monthly and weekly indices of the two Indian Famous stock exchanges BSE and NSE are reasonably Forecasted.

* 1. **CONCLUSION ON BSE**

From the monthly forecasted indices of BSE SENSEX , we may notice that once again the BSE SENSEX may cross the mark of 1,00,000 approximately . In the beginning of the year 2024 where as from the weekly forecasted indices, SENSEX may cross the mark of 1,00,000 points in the2nd quarter of the year 2023.The variation in the forecasted of monthly and weekly indices is due to the fact that we build the ARIMA model of monthly data pertaining to a long period 2013 – 2022,where as we build the ARIMA model of the weekly data pertaining to a short period 2018 – 2022.

* 1. **CONCLUSION ON NSE**

From the monthly forecasted indices of NSE NIFTY , we may notice that once again the NSE NIFTY may cross the mark of 82,000 approximately . In the beginning of the year 2024 where as from the weekly forecasted indices, NIFTY may cross the mark of 82,000 points in the2nd quarter of the year 2023.The variation in the forecasted of monthly and weekly indices is due to the fact that we build the ARIMA model of monthly data pertaining to a long period 2013 – 2022,where as we build the ARIMA model of the weekly data pertaining to a short period 2018 – 2022.

We may also notice the Synchronization between the forecasts of BSE SENSEX and NSE NIFTY indices.

The weekly forecast intervals are wider than the monthly forecast intervals, in other words, interval forcasting yields better forecast in case of monthly indices for a long period where as weekly indices for short period .

From the above graphs and tables, we can say that the indices of BSE SENSEX is increased and the indices of NSE NIFTY is also increased. when compared with the started year 2013 .

From the above graphs and tables ,we can say that the indices of BSE SENSEX can be reached to 1,00,000 at 2024.And the indices of NSE NIFTY can be reached to 82,000.

**5.3 FURTHER SCOPE**

The scope of further research and exploration related to the BSE and NSE, including aspects beyond modeling and forecasting using time series analysis in R:

**1. Behavioural Finance Analysis:**

Investigate the influence of investor behavior, emotions, and cognitive biases on stock market movements. Explore how psychological factors impact trading decisions and contribute to market anomalies.

**2. Market Microstructure Analysis:**

Delve into the mechanics of how orders are executed, the impact of trading volumes on prices, and the behavior of market participants. Analyze market liquidity, bid-ask spreads, and order flow dynamics.

**3. High-Frequency Trading Impact:**

Examine the impact of high-frequency trading on price volatility, liquidity, and market stability. Investigate the role of algorithmic trading in shaping intraday price patterns.

**4. Impact of News and Social Media:**

Study how news sentiment, social media discussions, and public sentiment affect stock prices. Utilize natural language processing techniques to analyze textual data.

**5. Market Efficiency and Anomalies:**

Evaluate the efficiency of the BSE and NSE in processing information and reflect on potential market anomalies, such as the January effect, momentum anomalies, or the weekend effect.

**6. International Market Linkages:**

Explore how global economic events and developments impact Indian stock markets. Analyze correlations and spillover effects between Indian indices and international markets**.**

**7. Sectoral and Industry Analysis:**

Focus on individual sectors or industries within the BSE and NSE. Investigate factors influencing sector-specific performance, and examine how economic trends impact different industries.

**8. Market Sentiment Index Creation:**

Develop composite sentiment indices that reflect investor sentiment based on a combination of news sentiment, social media activity, and market data.

**9. Market Network Analysis:**

Utilize network analysis techniques to understand the relationships between stocks, industries, and sectors within the BSE and NSE. Explore how disruptions in one sector can propagate across the network.

**10. Forecast Evaluation Methods:**

Research and implement advanced methods for evaluating forecasting models, including time-varying volatility, rolling-window analysis, and advanced performance metrics.

**11. Market Regulation and Policies:**

Examine the impact of regulatory changes, policy decisions, and economic reforms on stock market behavior. Assess how regulatory measures affect investor behavior and market stability.

**12. Long-Term Investment Strategies:**

Explore the effectiveness of long-term investment strategies like value investing or growth investing in the context of the BSE and NSE. Evaluate how these strategies perform over various economic cycles.

**13. Impact of Macroeconomic Indicators:**

Investigate the relationship between key macroeconomic indicators (GDP, inflation, interest rates) and stock market performance. Analyze how changes in economic fundamentals influence market trends.

**14. Environmental, Social, and Governance (ESG) Analysis:**

Study how ESG factors impact stock prices and investor preferences within the BSE and NSE. Examine the integration of sustainability criteria into investment decisions.

**15. Machine Learning for Predictive Analytics:**

Expand beyond time series analysis and utilize machine learning techniques for predictive analytics, sentiment analysis, and pattern recognition in stock market data.

**16. Real-Time Market Monitoring:**

Develop real-time monitoring systems that analyze market data as it becomes available, providing insights into emerging trends and potential anomalies.

**17. Investor Behaviour and Demographics:**

Investigate how demographic factors, investor profiles, and generational shifts influence investment decisions and market dynamics.

**18. Market Integration and Co-movement:**

Explore the degree of integration between BSE and NSE markets, analyzing co-movement of indices, cross-listed stocks, and the potential for arbitrage opportunities.

**19. Dynamic Asset Allocation Strategies:**

Design dynamic asset allocation strategies that adapt to changing market conditions, optimizing portfolio weights based on real-time data.

**20. Ethical and Sustainable Investing:**

Investigate the growth of ethical and sustainable investing in the BSE and NSE, examining how investors factor in social and environmental considerations.

Remember that each of these areas offers unique insights into the functioning of the BSE and NSE. Depending on your interests and expertise, you can contribute to a deeper understanding of financial markets, investor behavior, regulatory impact, and emerging trends.

5.4 REFERENCES:

1. **Anderson, T.W.(1971)**.The Statistical Analysis of Time Series, Willey New York.
2. **Bowerman, B.L and O’Connell,R . T.(1993).**Forecasting and Time Series : An Applied Approach , 3rd edition. Duxbury Press.
3. **Box , G E .P and Jenkins,G.M.(1994).** Time Series Analysis : Forecasting and Control , 3rd Edition.Prentice Hall.
4. **Box, G.E. P and Jenkins,G.M. (1968) .** some recent advances in Forecasting and Control 1, Application of Statistics , 17.
5. **Box, G.E .P and Jenkins,G.M. and D.W Bacon (1967).**Models for forecasting Seasonal and nonseasonal Time Series **J.D Cryer (1986).** Time Series Analysis .Duxbury Press.
6. **Box , G E .P and Jenkins,G.M.(1994).** Time Series Analysis :
7. Forecasting and control,revised edition**,Holden-Day,San Francisco.**
8. **Brockwell, P.J and Davix,R.A(1987)** Time Series : Theory and methods,Springer-Verlag, New York .
9. **Farnum , N.R and Stanton, L. W. (1989) .** Quantitative Forecasting Methods. PWS-Kent.
10. **Fuller,W.A (1976)** introduction to statistical Time series,Willey,New York.
11. **Gujarti,D.N (2004).** Basic Econometrics , TATA McGRAWHILL edition.
12. **Winter , P.R (1960)** “Forecastring sales of Eponentially Weighted Moving Average .