JINDAL GLOBAL BUSINESS SCHOOL

INDIA'S FIRST MULTI-DISCIPLINARY GLOBAL BUSINESS SCHOOL

Advanced Time Series Analytics

BS-BAT-645

Forecasting of Retail Sales Clothing stores in United States: An ARIMA based time series analysis

Group Members:

Neha Chowdary Ananthaneni – 21020366

Nunna Venkata Mahitha – 21020338

Under the guidance of Dr. Rajni

Date of Submission: 12th May 2025

Abstract:

The retail clothing sector in the United States reflects broader consumer behaviour and economic activity, showing significant seasonal patterns and long-term trends. This study investigates monthly retail sales data for clothing stores from January 2005 to December 2024, aiming to forecast sales for the next 12 months using ARIMA-based time series techniques. The analysis involves identifying trends, seasonal components, and fluctuations caused by economic events such as the COVID-19 pandemic. The study provides future sales forecasts with confidence intervals, which can help retailers and policymakers in planning inventory, logistics, and marketing strategies. The analysis considers only the total retail sales in the clothing segment for the United States.

INTRODUCTION

Retail sales serve as a key indicator of consumer behavior and economic activity in any country. In the United States, the clothing sector forms an important part of the retail landscape, contributing significantly to monthly retail trade statistics. This segment reflects both seasonality—driven by fashion cycles, weather changes, and holiday seasons—as well as underlying economic trends and consumer sentiment. In this study, we analyze monthly retail sales data for clothing stores from January 2005 to December 2024. The data is collected from the Federal Reserve Bank of St. Louis (FRED) under the series MRTSSM4481USN and is seasonally adjusted to eliminate recurring annual patterns. (Federal Reserve Bank of St. Louis, 2024)

Retail sales in this category peaked at over $29 billion in December 2021, a significant rise from approximately $15 billion in January 2005, indicating a near doubling over two decades. Sales tend to spike during the year-end holiday season (November–December), reflecting heightened consumer activity during that period. Conversely, lower values are typically observed in the first quarter of each year. These fluctuations underline the strong seasonal component inherent in apparel sales. The sector, like many others, experienced an abrupt decline in early 2020 due to the COVID-19 pandemic, with sales dropping to below $7 billion in April 2020 (Barron's, 2025), marking the sharpest contraction in the dataset. However, a rapid rebound followed in the subsequent months, showing resilience and recovery.

The U.S. retail clothing sector faces a mix of structural and cyclical challenges, including changing fashion preferences, the rise of e-commerce, and broader macroeconomic factors such as inflation and employment rates. Despite these challenges, the sector has continued to recover and grow, supported by digital transformation and adaptive business models. It is anticipated that online and omnichannel retailing will continue to reshape the landscape, influencing monthly sales trends and seasonality patterns.

This analysis aims to identify key components such as trend, seasonality, and residual variations in the retail sales time series. To accomplish this, the study utilizes the Autoregressive Integrated Moving Average (ARIMA) model—a widely used forecasting technique for univariate time series data. ARIMA helps forecast future sales by analyzing the past patterns and structure in the data, including autocorrelations and differencing to achieve stationarity. Several researchers have successfully applied ARIMA models in domains such as energy production, solar radiation forecasting, and economic modelling, making it suitable for retail applications as well.

This research uses the R programming platform to perform ARIMA modelling, employing packages such as forecast and ggplot2 for model fitting, visualization, and stationarity testing. The results from this study will provide monthly sales forecasts for the near future and insights into the behaviour of apparel retail over the past two decades. These findings can help retailers, analysts, and policymakers better understand demand cycles and plan more effectively for inventory, marketing, and logistics.

METHODOLOGY

Autoregressive Integrated Moving Average (ARIMA)

The ARIMA model is a widely accepted approach for time series forecasting, especially when the dataset exhibits both trend and seasonality (Hyndman & Athanasopoulos, 2018). It is a linear model that uses the past values of the series (autoregressive part), past forecast errors (moving average part), and differencing operations (to ensure stationarity) to model and forecast future data points. In this study, the ARIMA methodology is used to analyze and forecast monthly retail sales for clothing stores in the United States over the period January 2005 to December 2024.

The process begins by visually inspecting the raw time series to identify components such as trend and seasonality. A consistent upward trend is observed from 2005 to 2019, with a sharp dip in 2020 due to the COVID-19 pandemic, followed by a recovery. Clear seasonal fluctuations are also evident, particularly with repeated peaks around November and December each year. These observations suggest that a seasonal ARIMA (SARIMA) model would be appropriate for capturing both non-seasonal and seasonal dynamics in the data.

To build the ARIMA model, the Box–Jenkins methodology is followed, which involves four main steps: model identification, parameter estimation, diagnostic checking, and forecasting (Hyndman & Athanasopoulos, 2018). The raw data is first tested for stationarity using the Augmented Dickey-Fuller (ADF) and KPSS tests. Initial results indicate that the original time series is non-stationary due to both trend and seasonality. Therefore, seasonal differencing at lag 12 (to remove yearly cycles) and first-order differencing (to remove trend) are applied. The transformed data is found to be stationary based on statistical tests, confirming the suitability for ARIMA modelling.

The autocorrelation function (ACF) and partial autocorrelation function (PACF) plots of the differenced series are analyzed to select initial values for the ARIMA parameters (p, d, q) and seasonal parameters (P, D, Q). Multiple models are evaluated, and the best-performing one is selected using information criteria such as the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC). Additionally, the auto.arima() function in R is used to suggest optimal model parameters based on maximum likelihood estimation.

Once the model is fitted, residual diagnostics are performed to assess the adequacy of the model. The Ljung–Box test is used to verify that the residuals are white noise (i.e., they show no significant autocorrelation). Residual plots are also examined visually to check for randomness, constant variance, and normal distribution. If the residuals pass these diagnostic tests, the model is considered suitable for forecasting.

Finally, the trained ARIMA model is used to generate forecasts for the upcoming 12 months. These forecasts are presented along with 80% and 95% confidence intervals, offering a range of expected values that reflect the uncertainty in predictions. The model's accuracy is also assessed using performance metrics such as Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE), calculated on a test set derived from the most recent observations.

The plot of raw data for monthly retail sales of clothing stores in the U.S., as shown in Fig. 1, reveals clear components of both trend and seasonality in the time series. Seasonality is evident in the recurring peaks during the months of November and December each year, corresponding to the holiday shopping period. In contrast, dips are commonly observed in the early months of each year, particularly January and February. An increasing trend is also noticeable over the two-decade period, particularly after the 2008 financial crisis and again following the sharp decline during the COVID-19 pandemic in early 2020. The data, recorded in millions of U.S. dollars and adjusted for seasonality, reflects cumulative monthly sales across the retail clothing sector and serves as a basis for further time series modelling and forecasting.

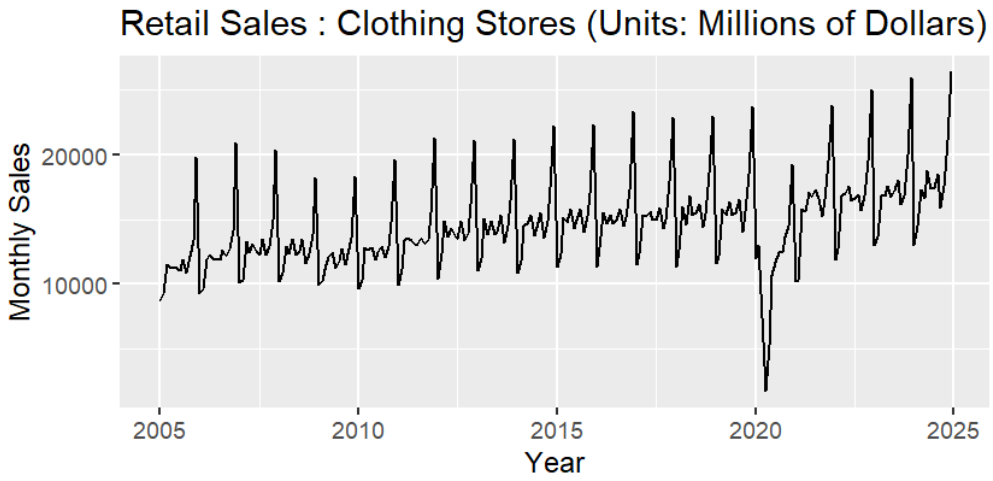

Fig 1: Raw Data of time series

Decomposition

Decomposition of the time series data helps in identifying and isolating the individual components—namely, trend, seasonality, and residuals—present in the retail sales data. As shown in Fig. 2, the seasonal decomposition of the monthly sales series reveals strong and consistent seasonality with pronounced peaks in November and December, which align with major retail events such as Black Friday and holiday shopping (Hyndman & Athanasopoulos, 2018). The trend component exhibits a generally upward trajectory from 2005 to 2024, despite temporary disruptions, notably during the 2008–2009 recession and the COVID-19 pandemic in 2020. These disruptions are reflected in the residual component as irregular spikes and dips. The presence of regular seasonal patterns and long-term growth supports the use of seasonal ARIMA (SARIMA) modelling for accurate forecasting. Visual inspection of the decomposition confirms the need for both seasonal and trend differencing to transform the data into a stationary series suitable for ARIMA modelling.

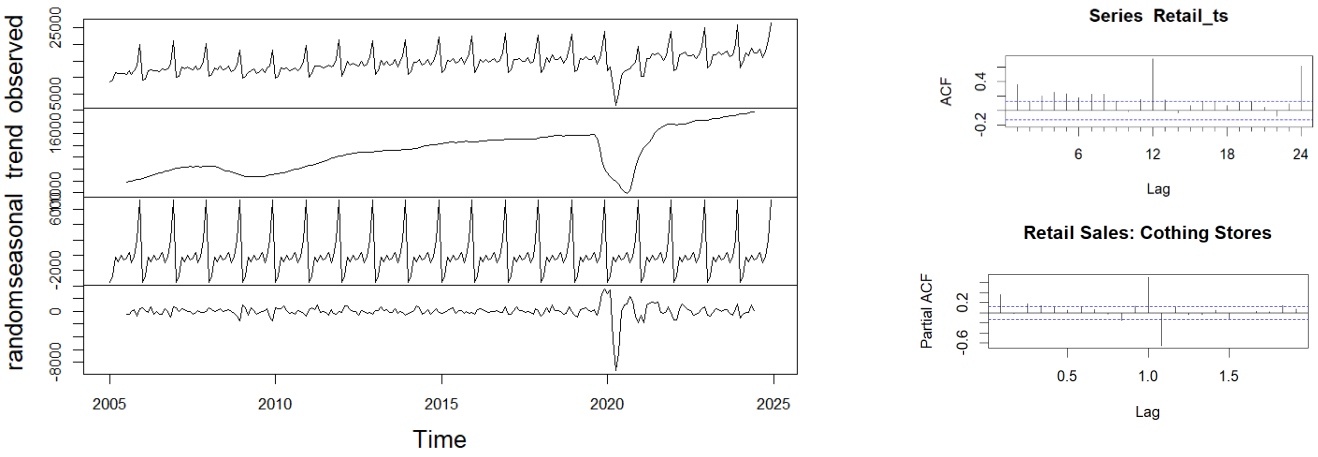

Fig 2: Decomposition and ACF and PACF

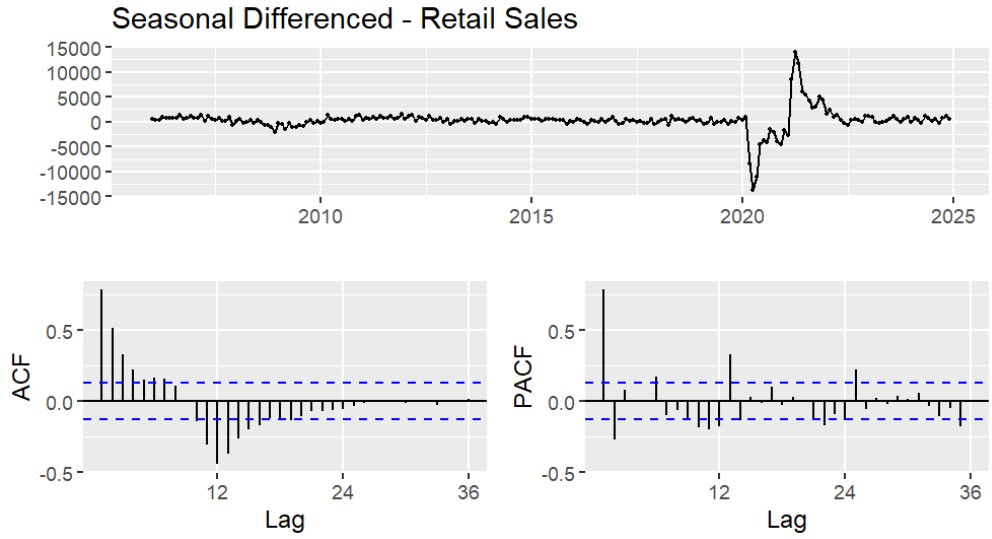

Fig 3: Seasonal Differenced time series data

Stationarity Test

For ARIMA modelling to be effective, the input time series must be stationary, meaning its statistical properties—such as mean and variance—should remain constant over time. To assess stationarity in the monthly retail sales data, two commonly used statistical tests are employed: the Augmented Dickey-Fuller (ADF) test and the Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test (Hyndman & Athanasopoulos, 2018).

ADF test:

- Null-Hypothesis: - “The time series is not stationary.”
- Alternate-Hypothesis – “The time series is stationary.”

KPSS test:

- Null-Hypothesis: - “The time series is stationary around a trend.”
- Alternate-Hypothesis – “The time series is not stationary around a trend.”

Test on Raw Series Data:

The ADF test on the original series returns a Dickey-Fuller statistic of -4.005 with a p-value of 0.01, indicating that the null hypothesis of non-stationarity can be rejected at the 5% significance level. Simultaneously, the KPSS test returns a test statistic of 0.1012 with a p-value of 0.1, suggesting that the null hypothesis of stationarity around a trend cannot be rejected. Together, these results imply that the raw series is reasonably stationary, though the presence of visible trend and seasonal components in the time series plot calls for further verification through differencing.

Test on Seasonally Differenced Data:

To eliminate visible seasonality, the data is first seasonally differenced using a lag of 12. The ADF test on this transformed series gives a Dickey-Fuller statistic of -4.1716 with a p-value of 0.01, confirming stationarity. The KPSS test reports a value of 0.05388 and a p-value of 0.1, which again suggests that the null hypothesis of trend stationarity cannot be rejected. These results indicate that the series is stable after seasonal differencing, though minor trend components may still persist.

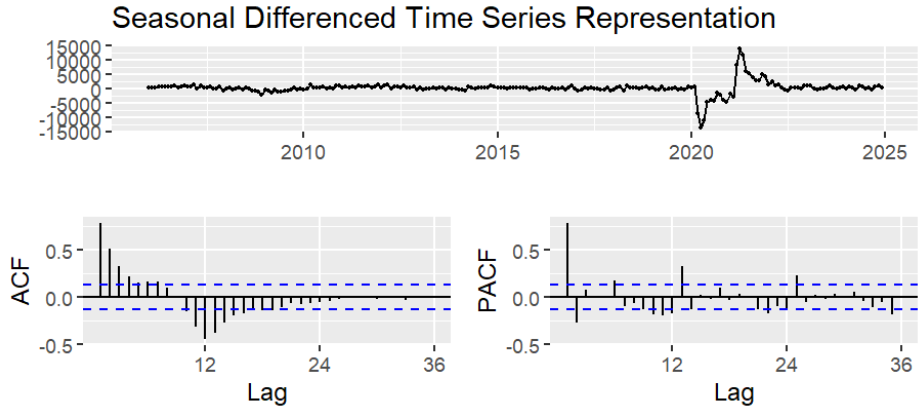

Fig 3: Seasonal Differenced time series data and its ACF and PACF plot

Test on Seasonally + First Differenced Data:

A second round of differencing is applied to the seasonally differenced data to account for the underlying linear trend, resulting in a series that is both seasonally and trend differenced. At this stage, the ADF test yields a Dickey-Fuller statistic of -5.7062 with a p-value less than 0.01, clearly indicating that the null hypothesis of non-stationarity can be rejected at the 5% significance level. The KPSS test on the same series returns a test statistic of 0.01293 with a p-value of 0.1, confirming that the null hypothesis of stationarity around a trend cannot be rejected. These results jointly confirm that the final transformed series is stationary and suitable for ARIMA modelling.

The transformation successfully eliminates systematic patterns, including trend and seasonality, thereby meeting the primary condition for effective ARIMA application. Fig. 3

shows the final differenced time series along with its corresponding ACF and PACF plots, which further validate the removal of persistent seasonal and trend components in the data.

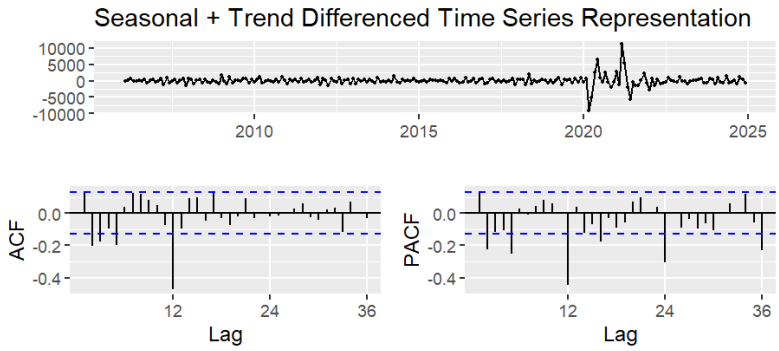

Fig 4: Seasonal plus Trend-differenced data and its ACF and PACF plot.

Initial ARIMA model selection

The ACF plot has significant spikes at lag 1, lag 2, lag 4 and lag 12 as shown in Fig.5. We can reject the spike on the higher-order lag, and the **q** can take a value of 1, 2 or 3. Similarly, the PACF plot has significant spikes at lag 1, lag 4, and lag 12, and the order of **p** can take a value of 1 or 2.

To identify of seasonal order: The ACF and PACF plot for the final differenced data for 36 lags is as shown in Fig. 5. The plot of ACF shows major spikes at seasonal lag 12, thus the **Q** can take the value of 1. Also, the **P** value can take the value 1 or 2 or 3 as in PACF we have significance at lag 12, lag 24 and lag 36.

To summarize, for the starting seasonal ARIMA model, we can propose:

- Non-seasonal part: (p, d, q) → p = 1 or 2, d=1 as per stationarity check, q = 1,2 or 3
- Seasonal part: (P, D, Q) → P = 1, D=1 as per seasonal differencing check, Q = 1, s=12

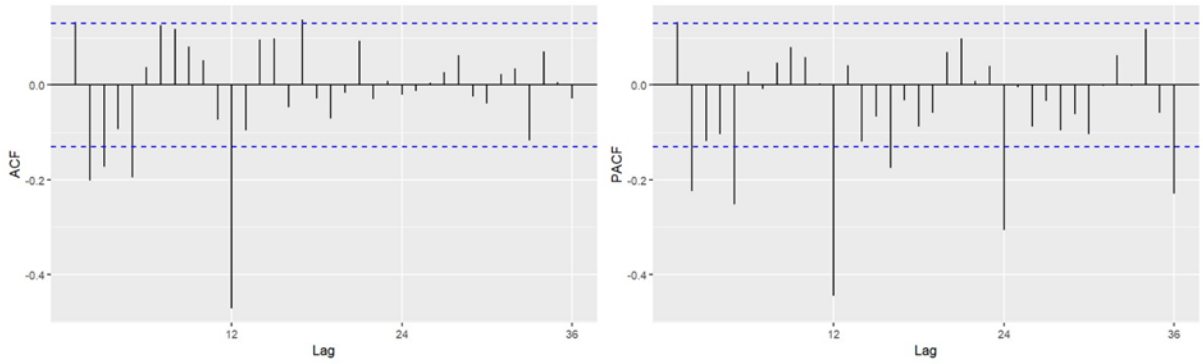

Fig 5: ACF and PACF plots of final differenced data.

Based on the analysis, the combinations of models in consideration are shown in Table 1 below.

Table 1: Initial models into consideration

S.no	h	d	D	P	p	Q	q	ARIMA MODEL
1	12	1	1	1	1	1	1	ARIMA (1,1,1)(1,1,1)(12)
2	12	1	1	1	1	1	2	ARIMA(1,1,2)(1,1,1)(12)
3	12	1	1	1	1	1	3	ARIMA(1,1,3)(1,1,1)(12)
4	12	1	1	1	2	1	1	ARIMA(2,1,1)(1,1,1)(12)
5	12	1	1	1	2	1	2	ARIMA(2,1,2)(1,1,1)(12)
6	12	1	1	1	2	1	3	ARIMA(2,1,3)(1,1,1)(12)
7	12	1	1	2	1	1	1	ARIMA(1,1,1)(2,1,1)(12)
8	12	1	1	2	1	1	2	ARIMA(1,1,2)(2,1,1)(12)
9	12	1	1	2	1	1	3	ARIMA(1,1,3)(2,1,1)(12)
10	12	1	1	2	2	1	1	ARIMA(2,1,1)(2,1,1)(12)
11	12	1	1	2	2	1	2	ARIMA(2,1,2)(2,1,1)(12)
12	12	1	1	2	2	1	3	ARIMA(2,1,3)(2,1,1)(12)
13	12	1	1	3	1	1	1	ARIMA(1,1,1)(3,1,1)(12)
14	12	1	1	3	1	1	2	ARIMA(1,1,2)(3,1,1)(12)
15	12	1	1	3	1	1	3	ARIMA(1,1,3)(3,1,1)(12)
16	12	1	1	3	2	1	1	ARIMA(2,1,1)(3,1,1)(12)
17	12	1	1	3	2	1	2	ARIMA(2,1,2)(3,1,1)(12)
18	12	1	1	3	2	1	3	ARIMA(2,1,3)(3,1,1)(12)

Using the principle of parsimony, we apply multiple ARIMA models derived from the combinations outlined in Table 1, starting with ARIMA (1,1,1) (1,1,1) (12) and subsequently exploring other configurations. For each model, the resulting AIC (Akaike Information Criterion) values are computed and compared to identify the best-fitting model, as summarized in Table 2. Based on the ranking of the AIC values, we then shortlist the top four ARIMA models for further analysis and evaluation.

Table 2: Selection of ARIMA model

S.no	ARIMA MODEL	AIC	Rank
1	ARIMA (1,1,1)(1,1,1)(12)	3832.12	13
2	ARIMA(1,1,2)(1,1,1)(12)	3890.72	15
3	ARIMA(1,1,3)(1,1,1)(12)	3810.42	6
4	ARIMA(2,1,1)(1,1,1)(12)	3811.58	10
5	ARIMA(2,1,2)(1,1,1)(12)	3810.85	9
6	ARIMA(2,1,3)(1,1,1)(12)	3810.1	4
7	ARIMA(1,1,1)(2,1,1)(12)	3832.25	14
8	ARIMA(1,1,2)(2,1,1)(12)	3808.71	2
9	ARIMA(1,1,3)(2,1,1)(12)	n/A	
10	ARIMA(2,1,1)(2,1,1)(12)	3811.98	11
11	ARIMA(2,1,2)(2,1,1)(12)	n/A	
12	ARIMA(2,1,3)(2,1,1)(12)	3810.22	5
13	ARIMA(1,1,1)(3,1,1)(12)	3831.49	12
14	ARIMA(1,1,2)(3,1,1)(12)	3808.22	1
15	ARIMA(1,1,3)(3,1,1)(12)	3810.47	7
16	ARIMA(2,1,1)(3,1,1)(12)	3810.51	8
17	ARIMA(2,1,2)(3,1,1)(12)	n/A	
18	ARIMA(2,1,3)(3,1,1)(12)	3809.28	3

Testing the ARIMA model

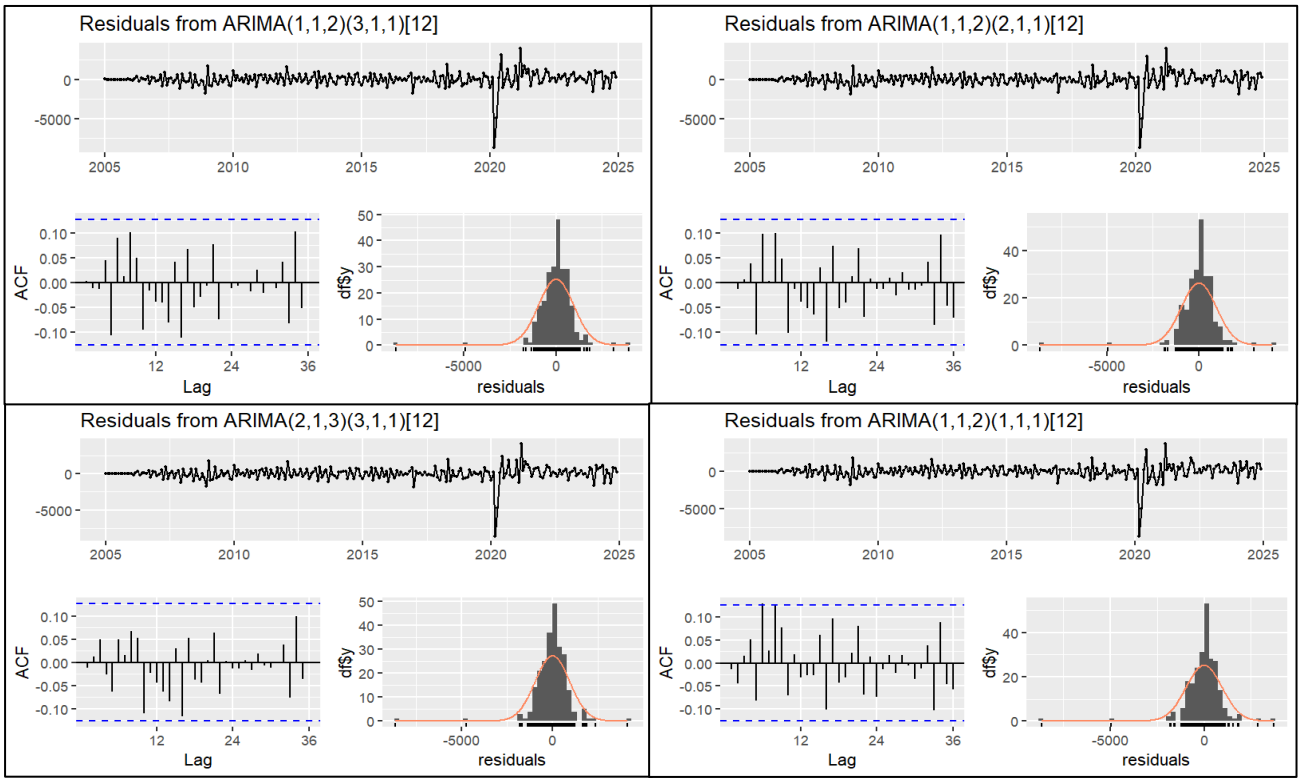

Fig 6: Residual plots for selected ARIMA models.

Ljung-Box Test is a statistical test to establish autocorrelations at multiple lags jointly.

- Null-Hypothesis: "There is no joint autocorrelation in the series up to lag k."
- Alternate-Hypothesis: "There is joint autocorrelation in the series when tested up to lag k."

The Ljung-Box test on residuals helps in identifying the fit of the model. The model with no autocorrelation is suitable as it captures all the information available in the data. Results of the Ljung –Box test on the models are as follows:

- For ARIMA (1,1,2) (3,1,1) (12): The p-value = 0.5016 > 0.05.
- For ARIMA (1,1,2) (2,1,1) (12) with p-value = 0.4729> 0.05
- ARIMA (2,1,3) (3,1,1) (12) with p-value = 0.7851> 0.05
- ARIMA (1,1,2) (1,1,1) (12) with p-value = 0.2756 i.e. > 0.05.

Hence all the null- hypothesis cannot be rejected. No autocorrelation is observed for residuals upto 12 lags, and the models are fit for forecasting.

Forecasting Accuracy

The time-series data is partitioned into the training data set and test data (last 13 data points). The training-set is utilized to train ARIMA model for forecasting, then it is tested on the test-data set to measure accuracy for forecasting. The forecasting accuracies of various models are in Table 3 (Bandara et al., 2019; Bi et al., 2020).

Table 3: The forecast accuracy for 4 selected ARIMA models

Type	RMSE	MAE	MAPE
ARIMA(1,1,2)(3,1,1)(12)			
Training	954.965	556.258	5.38903
Test	2002.29	1667.82	9.0731
ARIMA (1,1,2)(2,1,1)(12)			
Training	962.215	565.557	5.43784
Test	1104.29	937.996	4.9542
ARIMA(2,1,3)(3,1,1)(12)			
Training	948.638	556.847	5.34151
Test	1772.8	1447.08	7.8735
ARIMA(1,1,2)(1,1,1)(12)			
Training	952.511	560.075	5.3896
Test	1358.9	1177.49	6.1973

Following the RMSE criteria, ARIMA (1,1,2) (2,1,1) (12) performs the best on the test data set followed by ARIMA (1,1,2) (1,1,1) (12), and they can be used for prediction of the next periods. Using ARIMA (1,1,2) (2,1,1) (12) the forecasted value for the next 12 periods is displayed in Table 4.

Table 4: The forecasted value for the next 11 time periods.

Month	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
Jan-25	15034.3	13765.05	16303.55	13093.15	16975.45
Feb-25	15644.14	13815.24	17473.03	12847.09	18441.18
Mar-25	18059.72	16030.94	20088.51	14956.96	21162.48
Apr-25	17079.48	14950.49	19208.48	13823.46	20335.5
May-25	18328.06	16135.98	20520.14	14975.56	21680.55
Jun-25	17988.98	15749.84	20228.12	14564.51	21413.45
Jul-25	18328.67	16050.12	20607.21	14843.94	21813.4
Aug-25	18926.41	16612.41	21240.41	15387.45	22465.37
Sep-25	17411.81	15064.54	19759.07	13821.98	21001.64
Oct-25	18553.56	16174.34	20932.77	14914.86	22192.25
Nov-25	21039.74	18629.43	23450.05	17353.49	24725.98
Dec-25	26147.69	23706.91	28588.47	22414.84	29880.54

The values in Table 4 are plotted in Fig. 7. It displays the energy production along with the original data and forecasted data.

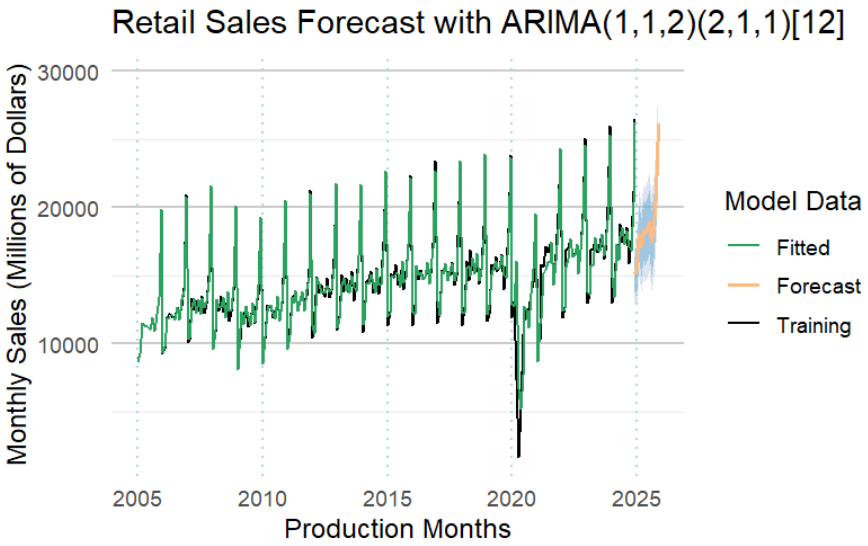

Fig 7: Forecasting using ARIMA for the United States Retail Sales

DISCUSSION

This study demonstrates the effectiveness of ARIMA modelling in forecasting monthly retail sales for clothing in the U.S., a sector marked by clear seasonality and long-term growth. Similar to the ICACE study by Rajni and Tuhin (Rajni & Tuhin, 2024), which applied ARIMA to forecast renewable energy production, our analysis used the Box–Jenkins methodology, incorporating seasonal and trend differencing to achieve stationarity. Both studies relied on ADF and KPSS tests to confirm data readiness for modelling, and the Ljung–Box test to ensure residuals were free of autocorrelation.

The strong seasonality observed—especially around holiday months—justified the use of a SARIMA model. Among various configurations, ARIMA (1,1,2) (2,1,1) (12) showed the best performance based on RMSE. While the ICACE study focused on energy, the consistent modelling approach across both domains underscores ARIMA's robustness in handling structured, cyclical data (Hyndman & Athanasopoulos, 2018). This supports its continued relevance for retail demand planning and economic forecasting.

CONCLUSION

This project utilized the United States Retail Sales: Clothing stores dataset to analyze and forecast monthly retail sales for clothing stores in the United States using ARIMA modelling. The data revealed strong seasonal patterns and long-term trends, which were effectively captured through decomposition and differencing techniques. After testing for stationarity and evaluating multiple model configurations, the selected seasonal ARIMA model demonstrated good forecasting accuracy and residual diagnostics, confirming its suitability.

The study also emphasizes the effectiveness of ARIMA in modelling real-world retail time series data and offers valuable insights into consumer behaviour and market trends. These findings can support strategic planning in areas such as inventory management, sales forecasting, and operational budgeting.

REFERENCES

1. Federal Reserve Bank of St. Louis. (2024). *Retail Sales: Clothing Stores (MRTSSM4481USN) [Seasonally adjusted, monthly, millions of dollars]*. Retrieved from https://fred.stlouisfed.org/series/MRTSSM4481USN
2. U.S. Census Bureau. (2024). *Monthly Retail Trade Report.* Retrieved from https://www.census.gov/retail/
3. Barron's. (2025, February 14). *January Retail Sales Slide. Consumers Tighten Their Belts*. Retrieved from https://www.barrons.com/articles/retail-sales-stocks-january-de50df59
4. Bi, X., Adomavicius, G., Li, W., & Qu, A. (2020). *Improving Sales Forecasting Accuracy: A Tensor Factorization Approach with Demand Awareness*. arXiv preprint arXiv:2011.03452. Retrieved from https://arxiv.org/abs/2011.03452
5. Bandara, K., Shi, P., Bergmeir, C., Hewamalage, H., Tran, Q., & Seaman, B. (2019). *Sales Demand Forecast in E-commerce using a Long Short-Term Memory Neural Network Methodology*. arXiv preprint arXiv:1901.04028. Retrieved from https://arxiv.org/abs/1901.04028
6. Li, C., Cheang, B., Luo, Z., & Lim, A. (2020). *An Exponential Factorization Machine with Percentage Error Minimization to Retail Sales Forecasting*. arXiv preprint arXiv:2009.10619. Retrieved from https://arxiv.org/abs/2009.10619
7. Chen, C., Liu, Z., Zhou, J., Li, X., Qi, Y., Jiao, Y., & Zhong, X. (2020). *How Much Can A Retailer Sell? Sales Forecasting on Tmall*. arXiv preprint arXiv:2002.11940. Retrieved from https://arxiv.org/abs/2002.11940
8. Effendi, S. Y. (2023, December 7). *ARIMA for Time-Series Forecasting: Retail Sales Predictions.* Medium. https://medium.com/@syarifyusuf/arima-for-time-series-forecasting-retail-sales-predictions-6e0da74232a0
9. Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice* (2nd ed.). OTexts. https://otexts.com/fpp2/
10. International Journal of Information Management Data Insights. (2022). *Time-series forecasting of seasonal items sales using machine learning – A comparative analysis*. Elsevier. Retrieved from https://www.sciencedirect.com/science/article/pii/S2667096822000027
11. Atlantis Press. (2020). *A Sales Forecasting Model for the Consumer Goods with Holiday Effects*. Journal of Risk Analysis and Crisis Response. Retrieved from https://www.atlantis-press.com/journals/jracr/125941835/view
12. Rajni, Banerjee T., & Kumar, P. (2024). Forecasting of renewable energy production in the United States: An ARIMA-based time series analysis. *AIP Conference Proceedings, 3010*, 030014. https://doi.org/10.1063/5.0193938
13. MDPI. (2022). *Enhancing Hierarchical Sales Forecasting with Promotional Data: A Comparative Study Using ARIMA and Deep Neural Networks*. Journal of Data. Retrieved from https://www.mdpi.com/2504-4990/6/4/128

APPENDIX

Step	R Code	Description
Load necessary libraries	library(fpp2) library(tseries)	Load libraries for time series analysis and forecasting.
Load the dataset	Retail_Sales <- read.csv("Retail_Sales.csv", header = TRUE)	Import the retail sales data from a CSV file.
Create time series object	Retail_ts <- ts(Retail_Sales$Sales, start = c(2005,1), frequency = 12)	Convert the sales data into a monthly time series starting in January 2005.
Check data structure	str(Retail_Sales)	Display the structure of the dataset.
Generate summary statistics	summary(Retail_Sales)	View summary statistics (min, max, mean, etc.) of the dataset.
Check for missing values	sum(is.na(Retail_Sales$Sales))	Count the number of missing values in the Sales column.
Plot time series	autoplot(Retail_ts) + ggtitle("Retail Sales : Clothing Stores (Units: Millions of Dollars) ") + ylab("Monthly Sales") + xlab("Year")	Create a basic time series line plot with title and axis labels.
Seasonal Plot	ggseasonplot(Retail_ts, year.labels = TRUE, year.labels.left = TRUE) + ggtitle("Seasonal Plot")	Create a seasonal plot to observe patterns repeating at the same time each year.
Seasonal Subseries Plot	ggsubseriesplot(Retail_ts) + ggtitle("Seasonal Subseries Plot")	Display subseries plots for each month to analyze seasonality more precisely.
Lag Plot	gglagplot(Retail_ts) + ggtitle("Lag Plot")	Visualize correlation between lagged values to detect patterns or randomness.
Autocorrelation Function	Acf(Retail_ts) + ggtitle("Retail Sales: Cothing Stores")	Plot the ACF to examine autocorrelation in the time series data.
Partial ACF Plot (basic)	pacf(Retail_ts, main = "Retail Sales: Cothing Stores")	Generate a partial autocorrelation plot to identify significant lags after controlling for others.
Partial ACF Plot (custom)	pacf(Retail_ts, lags = 60, main = "Partial ACF of Raw Retail Sales Data")	Create a PACF plot with custom number of lags (up to 60) for deeper analysis.
Decomposition of Time Series	decompose(Retail_ts) plot(decompose(Retail_ts))	Decompose the time series into trend, seasonal, and random components, and plot the result.
Linear Regression Model	reg_model <- tslm(Retail_ts ~ trend + season) summary(reg_model)	Fit a linear regression model to the time series with trend and seasonality as predictors.
Actual vs Fitted Plot	autoplot(Retail_ts) + autolayer(fitted(reg_model), series = "Fitted (Regression)") + ggtitle("Linear Regression Model: Retail Sales") + ylab("Sales") + xlab("Year")	Plot the actual retail sales against the fitted values from the regression model.
Simple Moving Average (4-Month)	ma_4 <- ma(Retail_ts, order = 4) autoplot(Retail_ts) + autolayer(ma_4, series = "4-Month MA") + ggtitle("Simple Moving Average (4-Month)") + ylab("Sales") + xlab("Year")	Calculate and plot a 4-month moving average to smooth the time series.
Simple Exponential Smoothing (SES)	ses_model <- ses(Retail_ts, h = 12) autoplot(ses_model) + ggtitle("Simple Exponential Smoothing")	Apply SES to forecast sales 12 months ahead and visualize the result.

Holt's Linear Trend Method	holt_model <- holt(Retail_ts, h = 12) autoplot(holt_model) + ggtitle("Holt's Linear Trend Smoothing")	Apply Holt's method considering trend and forecast the next 12 months.
Holt-Winters Additive Model	hw_add <- hw(Retail_ts, seasonal = "additive", h = 12) autoplot(hw_add) + ggtitle("Holt-Winters (Additive)")	Use Holt-Winters additive model to forecast with both trend and seasonality for 12 months ahead.
ADF Test	adf.test(Retail_ts, k=12)	Perform Augmented Dickey-Fuller test to check for unit root (non-stationarity) with lag = 12.
KPSS Test	kpss.test(Retail_ts, null="Trend")	Conduct KPSS test for stationarity around a trend (null hypothesis = trend-stationary).
Number of Differences (Initial)	ndiffs(Retail_ts)	Identify the number of non-seasonal differences needed to make the series stationary.
Seasonal Differencing	Retail_ts_seasonaldiff = diff(Retail_ts,12)	Apply seasonal differencing with a lag of 12 to remove seasonal pattern.
ACF and PACF after Seasonal Differencing	ggtsdisplay(Retail_ts_seasonaldiff, main= "Seasonal Differenced Time Series Representation")	Display the differenced time series along with ACF and PACF plots to check remaining autocorrelation
Number of Differences (Post-Seasonal)	ndiffs(Retail_ts_seasonaldiff)	Recheck number of non-seasonal differences needed after applying seasonal differencing.
ADF Test (Post-Seasonal Differencing)	adf.test(Retail_ts_seasonaldiff, k=12)	Check for stationarity using ADF test after seasonal differencing (lag = 12).
KPSS Test (Post-Seasonal Differencing)	kpss.test(Retail_ts_seasonaldiff, null="Trend")	Use KPSS test to validate if trend-stationarity remains after seasonal differencing.
Final Differencing	Retail_ts_Finaldiff = diff(Retail_ts_seasonaldiff, 1)	Apply additional non-seasonal differencing to remove trend components, ensuring full stationarity.
Visual Inspection	ggtsdisplay(Retail_ts_Finaldiff, main= "Seasonal + Trend Differenced Time Series Representation")	Display the differenced time series with ACF and PACF plots to visually assess residual autocorrelation.
ADF Test (Final)	adf.test(Retail_ts_Finaldiff, k=12)	Final ADF test to confirm full stationarity of the series.
KPSS Test (Final)	kpss.test(Retail_ts_Finaldiff, null="Trend")	Final KPSS test to confirm removal of trend and seasonal components.
Auto ARIMA Model Selection	auto.arima(Retail_ts_Finaldiff, trace = TRUE, approximation = FALSE)	Automatically identify the best ARIMA model parameters based on AIC/BIC by evaluating multiple configurations.
Fit multiple SARIMA models	arima(Retail_ts,order =c(1,1,1), seasonal = list(order = c(1,1,1), period = 12)) arima(Retail_ts,order =c(1,1,2), seasonal = list(order = c(1,1,1), period = 12)) arima(Retail_ts,order =c(1,1,3), seasonal = list(order = c(1,1,1), period = 12)) arima(Retail_ts,order =c(2,1,1), seasonal = list(order = c(1,1,1), period = 12)) arima(Retail_ts,order =c(2,1,2), seasonal = list(order = c(1,1,1), period = 12)) arima(Retail_ts,order =c(2,1,3), seasonal = list(order = c(1,1,1), period = 12)) arima(Retail_ts,order =c(1,1,1), seasonal = list(order = c(2,1,1), period = 12))	Fits a range of Seasonal ARIMA models to capture both non-seasonal and seasonal dynamics with seasonal period = 12. Variants explore different AR (p), MA (q), seasonal AR (P), and seasonal MA (Q) to identify best fit via model diagnostics like AIC/BIC.

	arima(Retail_ts,order =c(1,1,2), seasonal = list(order = c(2,1,1), period = 12)) arima(Retail_ts,order =c(1,1,3), seasonal = list(order = c(2,1,1), period = 12)) arima(Retail_ts,order =c(2,1,1), seasonal = list(order = c(2,1,1), period = 12)) arima(Retail_ts,order =c(2,1,2), seasonal = list(order = c(2,1,1), period = 12)) arima(Retail_ts,order =c(2,1,3), seasonal = list(order = c(2,1,1), period = 12)) arima(Retail_ts,order =c(1,1,1), seasonal = list(order = c(3,1,1), period = 12)) arima(Retail_ts,order =c(1,1,2), seasonal = list(order = c(3,1,1), period = 12)) arima(Retail_ts,order =c(1,1,3), seasonal = list(order = c(3,1,1), period = 12)) arima(Retail_ts,order =c(2,1,1), seasonal = list(order = c(3,1,1), period = 12)) arima(Retail_ts,order =c(2,1,2), seasonal = list(order = c(3,1,1), period = 12)) arima(Retail_ts,order =c(2,1,3), seasonal = list(order = c(3,1,1), period = 12))	
Automatically identify best ARIMA model	auto.arima(Retail_ts, approximation = FALSE)	Use auto.arima to select the best model based on AIC/BIC without using approximations.
Split data into training and test sets	training <- subset(Retail_ts, end=length(Retail_ts)-14) test <- subset(Retail_ts, start=length(Retail_ts)-13)	Divide the series to reserve the last 14 values for out-of-sample testing.
Model 1: ARIMA(1,1,2)(3,1,1)[12]	Model1 = arima(Retail_ts, order = c(1,1,2), seasonal = list(order = c(3,1,1), period = 12)) Box.test(Model_1_residuals, lag = 12, type = "Ljung-Box") checkresiduals(Model1) adf.test(Model_1_residuals) Model_test1 = arima(training, order = c(1,1,2), seasonal = list(order = c(3,1,1), period = 12)) forecast_ts1 = forecast(Model_test1, h = 13) AC_1 = accuracy(forecast_ts1, test)	Fit SARIMA model, check residual diagnostics, test stationarity, forecast next 13 points, and evaluate forecast accuracy.
Model 2: ARIMA(1,1,2)(2,1,1)[12]	Model2 = arima(Retail_ts, order = c(1,1,2), seasonal = list(order = c(2,1,1), period = 12)) Box.test(Model_2_residuals, lag = 12, type = "Ljung-Box") checkresiduals(Model2) adf.test(Model_2_residuals) Model_test2 = arima(training, order = c(1,1,2), seasonal = list(order = c(2,1,1), period = 12)) forecast_Ts2 = forecast(Model_test2, h = 13) AC_2 = accuracy(forecast_Ts2, test)	Fit SARIMA model, check residual diagnostics, test stationarity, forecast next 13 points, and evaluate forecast accuracy.
Model 3: ARIMA(2,1,3)(3,1,1)[12]	Model3 = arima(Retail_ts, order = c(2,1,3), seasonal = list(order = c(3,1,1), period = 12)) Box.test(Model_3_residuals, lag = 12, type = "Ljung-Box") checkresiduals(Model3) adf.test(Model_3_residuals) Model_test3 = arima(training, order = c(2,1,3), seasonal = list(order = c(3,1,1), period = 12)) forecast_Ts3 = forecast(Model_test3, h = 13) AC_3 = accuracy(forecast_Ts3, test)	Fit SARIMA model, check residual diagnostics, test stationarity, forecast next 13 points, and evaluate forecast accuracy.

Model 4: ARIMA(1,1,2)(1,1,1)[12]	Model4 = arima(Retail_ts, order = c(1,1,2), seasonal = list(order = c(1,1,1), period = 12)) Box.test(Model_4_residuals, lag = 12, type = "Ljung-Box") checkresiduals(Model4) adf.test(Model_4_residuals) Model_test4 = arima(training, order = c(1,1,2), seasonal = list(order = c(1,1,1), period = 12)) forecast_Ts4 = forecast(Model_test4, h = 13) AC_4 = accuracy(forecast_Ts4, test)	Fit SARIMA model, check residual diagnostics, test stationarity, forecast next 13 points, and evaluate forecast accuracy.
Identify Best Model Based on RMSE	comparison <- rbind(...)\nprint(round(comparison, 4))\nrmse_values <- comparison[, "RMSE"]\nbest_model_index1 <- which.min(rmse_values)\nbest_model_name1 <- rownames(comparison)[best_model_index1]\ncat("Best ARIMA model based on RMSE is:", best_model_name1)	Compare forecast accuracy metrics (RMSE, MAE, MAPE) across models and identify the best model based on lowest RMSE.
Identify Best Model Based on MAE	mae_values <- comparison[, "MAE"] best_model_index2 <- which.min(mae_values) best_model_name2 <- rownames(comparison)[best_model_index2] cat("Best ARIMA model based on MAE is:", best_model_name2, "\n")	
Identify Best Model Based on MAPE	mape_values <- comparison[, "MAPE"] best_model_index3 <- which.min(mape_values) best_model_name3 <- rownames(comparison)[best_model_index3] cat("Best ARIMA model based on MAPE is:", best_model_name3, "\n")	
Identify Overall Best Model	best_model_name <- "ARIMA (1,1,2)(2,1,1)[12]" cat("Best ARIMA model based on evaluation metrics (lowest RMSE, MAPE, etc.):", best_model_name, "\n")	
Generate Final Forecast with Confidence Intervals	forecast_final = forecast(arima(Retail_ts, order = c(1,1,2), seasonal = list(order = c(2,1,1), period = 12)), h = 12, level = c(80, 95))	Generate 12-step ahead forecast with 80% and 95% confidence intervals.
Load Required Libraries	library(forecast) library(ggplot2) library(zoo)	Load required libraries: forecast for time series modeling, ggplot2 for visualization, and zoo for handling time series dates.
Prepare Date Index from Time Series	dates <- as.Date(as.yearmon(time(Retail_ts)))	Convert the time series index into a Date format for monthly plotting.
Extract Fitted Values from Final Model	fitted_vals <- fitted(forecast_final$model)	Extract in-sample fitted values from the selected ARIMA model for comparison with actual training data.
Create DataFrame for Training and Fitted Values	train_df <- data.frame(Date = dates, Training = as.numeric(Retail_ts), Fitted = fitted_vals)	Combine the original training data and fitted values into a single data frame for plotting.
Generate Forecast Date Index	fc_dates <- seq(as.Date(as.yearmon(max(dates))) + 1/12, by = "month", length.out = length(forecast_final$mean))	Create a monthly sequence of forecast dates starting after the last training observation.
Create Forecast DataFrame with Confidence Intervals	fc_df <- data.frame(Date = fc_dates, Forecast = as.numeric(forecast_final$mean), Lo80 = forecast_final$lower[,"80%"], Hi80 = forecast_final$upper[,"80%"], Lo95 = forecast_final$lower[,"95%"], Hi95 = forecast_final$upper[,"95%"])	Create a data frame containing the forecasted values and corresponding 80% and 95% confidence intervals.
Generate Final Forecast Plot with Confidence Intervals	# Final Plot with lines ggplot() + # Confidence Intervals	Visualize the results by plotting training data, fitted values, forecast, and shaded confidence intervals using

	geom_ribbon(data = fc_df, aes(x = Date, ymin = Lo95, ymax = Hi95), fill = "#a6bddb", alpha = 0.4) + geom_ribbon(data = fc_df, aes(x = Date, ymin = Lo80, ymax = Hi80), fill = "#67a9cf", alpha = 0.5) + # Forecast Line geom_line(data = fc_df, aes(x = Date, y = Forecast, color = "Forecast"), size = 0.8) + # Training and Fitted Lines geom_line(data = train_df, aes(x = Date, y = Training, color = "Training"), size = 0.6) + geom_line(data = train_df, aes(x = Date, y = Fitted, color = "Fitted"), size = 0.6) + # Styling scale_color_manual(name = "Model Data", values = c("Training" = "black", "Fitted" = "#2ca25f", "Forecast" = "#fdbb84")) + labs(title = "Retail Sales Forecast with ARIMA(1,1,2)(2,1,1)[12]", x = "Production Months", y = "Monthly Sales (Millions of Dollars)") + theme_minimal(base_size = 12) + theme(panel.grid.major.x = element_line(color = "lightblue", linetype = "dotted"), panel.grid.minor.x = element_blank(), panel.grid.major.y = element_line(color = "grey80"), legend.position = "right")	ggplot2. Includes custom legends and styling for clarity.