The Impact of Economic Indicators on Luxury Market Growth: An ARIMAX Model Analysis and Forecast

zs

2024-12-06

Contents

```r
# Load required packages
library(tidyverse)   # For data manipulation and visualization
library(corrplot)    # For correlation matrix plotting
library(forecast)    # For time series forecasting methods
library(lmtest)
library(changepoint) # For structural break analysis (not extensively used here, but can be helpful)
library(zoo)         # For handling time series data
library(tseries)     # For stationarity tests (ADF)
library(xts)
```

```r
# Import the merged dataset containing luxury market growth and related indicators.
merged_data <- read.csv("merged_luxury_indicators.csv")

# Extract only the Year and Growth_Rate columns, and filter out missing values
luxury_points <- merged_data %>%
  dplyr::select(Year, Growth_Rate) %>%
  filter(!is.na(Growth_Rate))
```

```r
# Print the range of years available in the dataset
print("Data time range:")
```

```
## [1] "Data time range:"
```

```r
print(range(merged_data$Year))
```

```
## [1] 2004 2022
```

```r
# Print how many observations are available for each year
print("Number of observations per year:")
```

```
## [1] "Number of observations per year:"
```

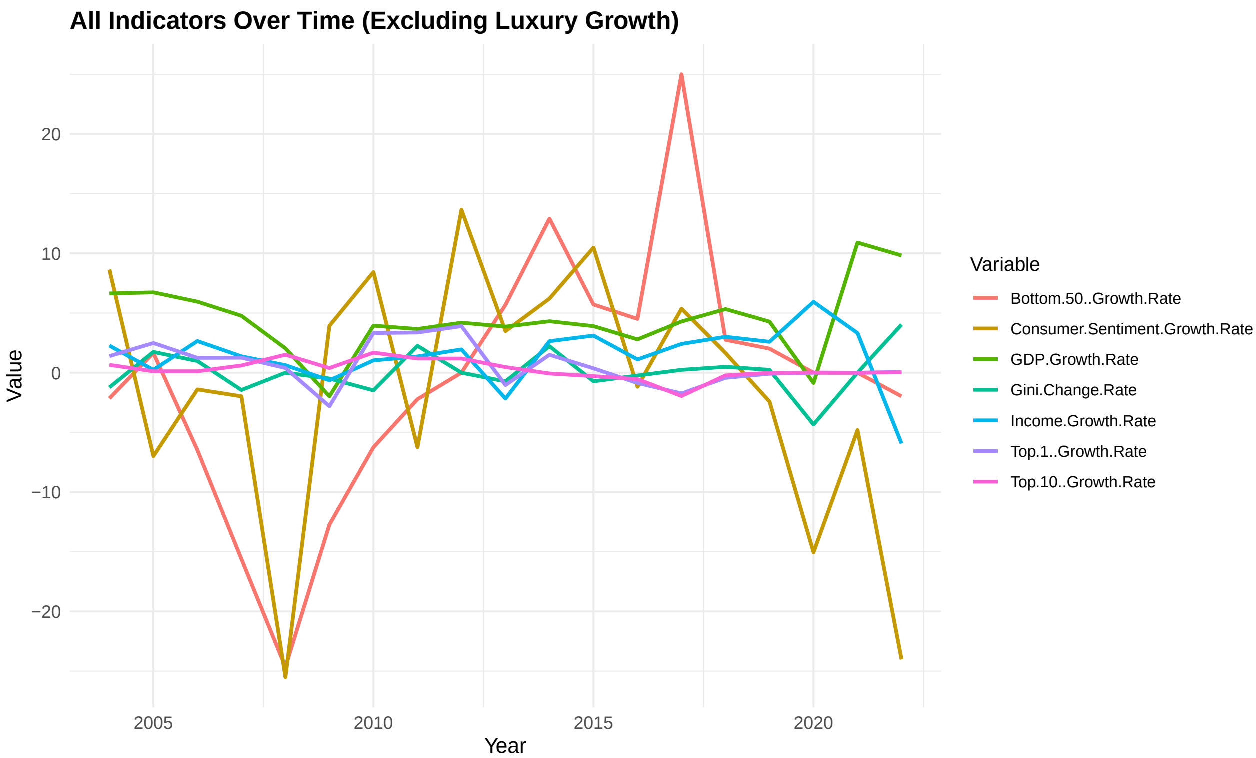
```r
print(table(merged_data$Year))
```

```
##
## 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019
##    6    6    6    6    6    6    8    9    9   10   10   10   10   10   10   10
## 2020 2021 2022
##   10   10   10
```

```r
# Check the structure of the luxury_points data frame to ensure proper data preparation
str(luxury_points)
```

```
## 'data.frame':    162 obs. of  2 variables:
##  $ Year       : int  2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 ...
##  $ Growth_Rate: num  33.33 8.87 -4.44 28.68 18.67 ...
```

```r
# Pivot the dataset from wide to long format, making it easier to plot multiple indicators over time
merged_long <- merged_data %>%
  tidyr::pivot_longer(
    cols = -Year,
    names_to = "Variable",
    values_to = "Value"
  )

# Filter out the main growth rate (we only want to plot the other indicators)
x_only_data <- merged_long %>%
  filter(Variable != "Growth_Rate")

# Plot all indicators (except Growth_Rate) over time
ggplot(x_only_data, aes(x = Year, y = Value, color = Variable)) +
  geom_line(linewidth = 1) +
  labs(
    title = "All Indicators Over Time (Excluding Luxury Growth)",
    x = "Year",
    y = "Value",
    color = "Variable"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10)
  )
```




```r
# Save the plot for reference
ggsave("x_variables_only_plot.pdf", width = 12, height = 6)
```

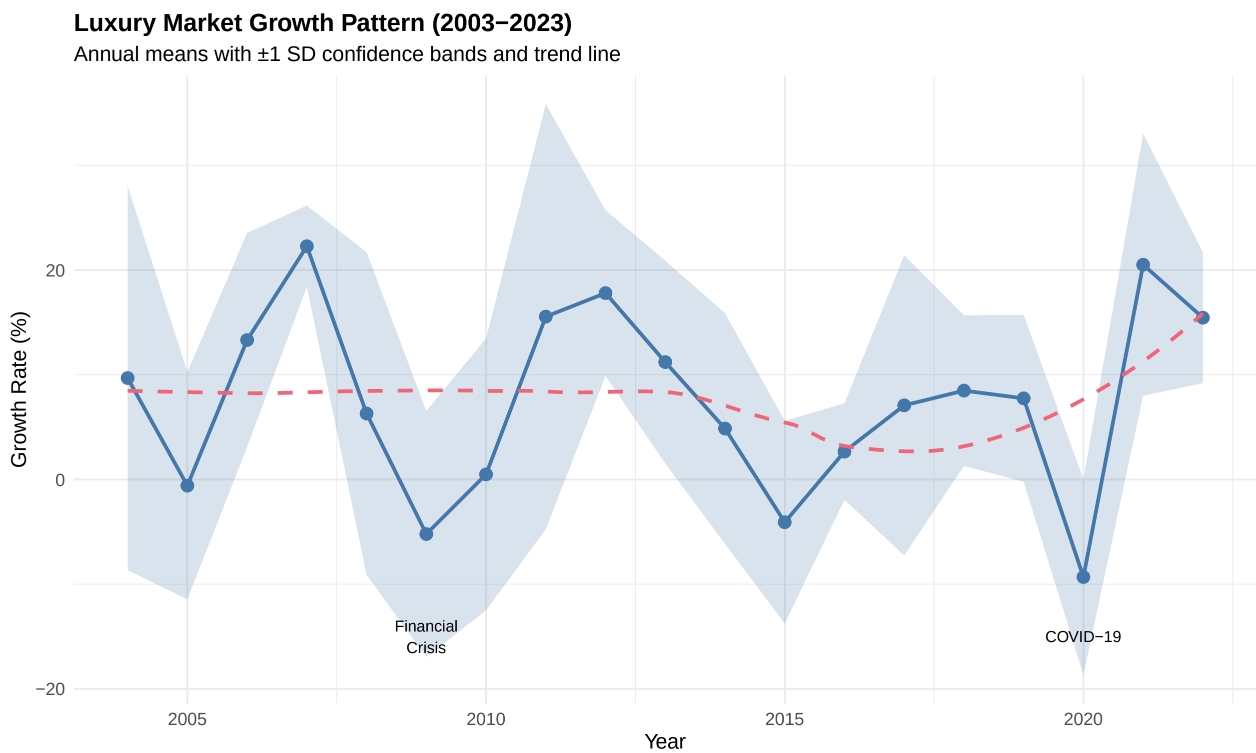
```r
# Compute annual mean and standard deviation of the Luxury Growth_Rate
yearly_pattern <- luxury_points %>%
  group_by(Year) %>%
  summarise(
    Mean = mean(Growth_Rate, na.rm = TRUE),
    SD = sd(Growth_Rate, na.rm = TRUE),
    Upper = Mean + SD,
    Lower = Mean - SD,
    .groups = 'drop'
  )

print(head(yearly_pattern))
```

```
## # A tibble: 6 x 5
##    Year   Mean    SD Upper  Lower
##   <int>  <dbl> <dbl> <dbl>  <dbl>
## 1  2004  9.68  18.4  28.0   -8.67
## 2  2005 -0.593 10.9  10.3  -11.4
## 3  2006 13.3   10.2  23.5    3.08
## 4  2007 22.3    3.89 26.2   18.4
## 5  2008  6.29  15.4  21.7   -9.12
## 6  2009 -5.21  11.8   6.54 -17.0
```

```r
# Visualize the pattern of luxury market growth over time with mean and ±1 SD bands
ggplot(yearly_pattern, aes(x = Year)) +
  geom_ribbon(aes(ymin = Lower, ymax = Upper),
              fill = "#4477AA",
              alpha = 0.2) +
```

```r
  geom_line(aes(y = Mean),
            color = "#4477AA",
            linewidth = 1) +
  geom_point(aes(y = Mean),
             color = "#4477AA",
             size = 3) +
  # Annotate known significant economic events
  annotate("text", x = 2009, y = -15,
           label = "Financial\nCrisis", size = 3) +
  annotate("text", x = 2020, y = -15,
           label = "COVID-19", size = 3) +
  # Add a smoothed trend line for long-term patterns
  geom_smooth(aes(y = Mean),
              method = "loess",
              color = "#EE6677",
              se = FALSE,
              linetype = "dashed") +
  labs(title = "Luxury Market Growth Pattern (2003-2023)",
       subtitle = "Annual means with ±1 SD confidence bands and trend line",
       x = "Year",
       y = "Growth Rate (%)") +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    plot.subtitle = element_text(size = 12),
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10)
  )
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



```r
ggsave("luxury_growth_pattern.pdf", width = 12, height = 6)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```r
# Basic data checks: the range of years, growth rate values, and number of data points
cat("Data range checks:\n")
```

```
## Data range checks:
```

```r
cat("Year range:", range(luxury_points$Year), "\n")
```

```
## Year range: 2004 2022
```

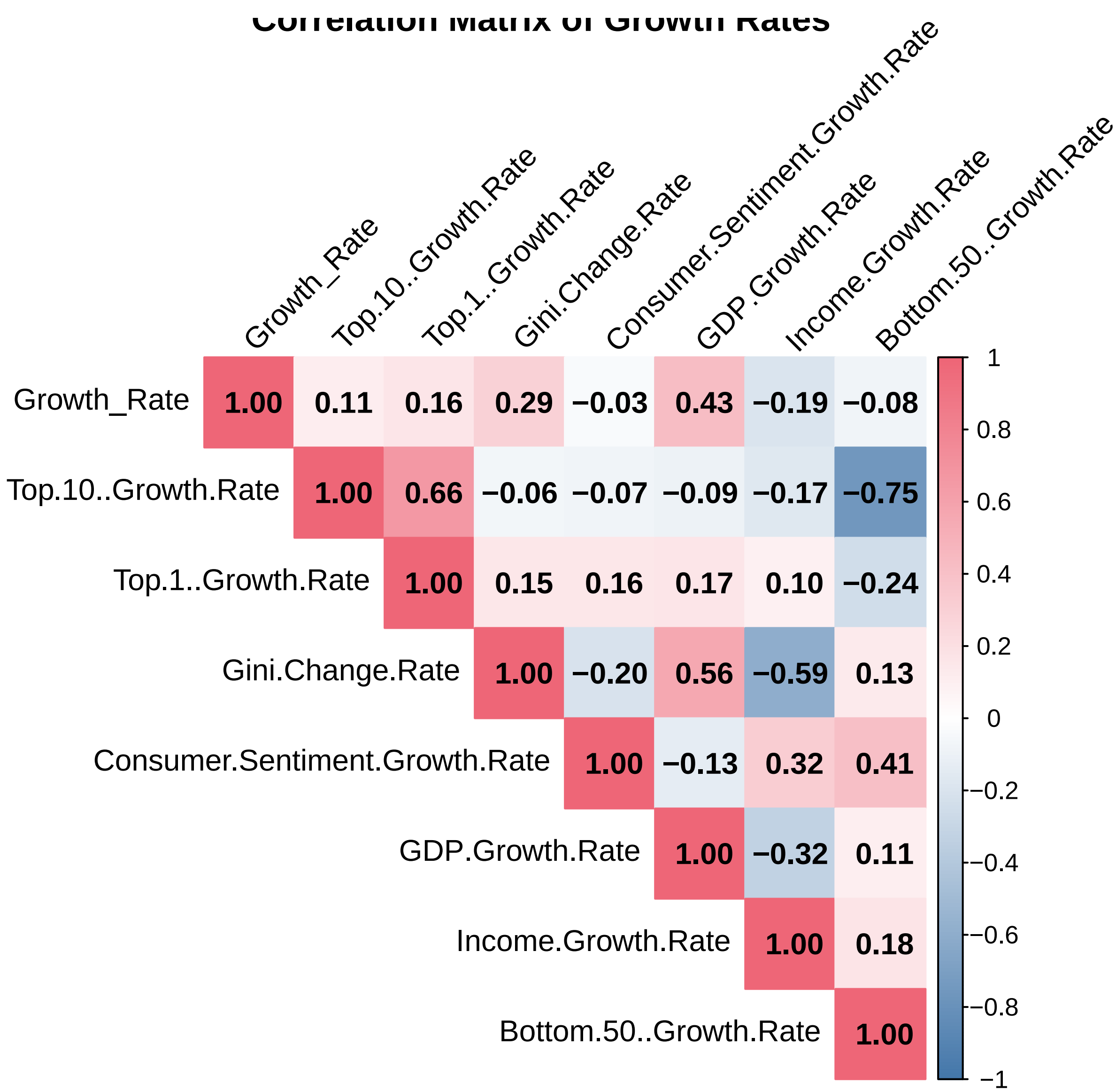
```r
cat("Growth rate range:", range(luxury_points$Growth_Rate), "\n")
```

```
## Growth rate range: -31.16883 39.19886
```

```r
cat("Number of data points:", nrow(luxury_points), "\n")
```

```
## Number of data points: 162
```

```r
# Calculate correlation among selected indicators
selected_cols <- c("Growth_Rate", "Top.10..Growth.Rate", "Top.1..Growth.Rate",
                   "Gini.Change.Rate", "Consumer.Sentiment.Growth.Rate",
                   "GDP.Growth.Rate", "Income.Growth.Rate", "Bottom.50..Growth.Rate")

correlation_matrix <- cor(merged_data[, selected_cols],
                          use = "pairwise.complete.obs")

# Plot a correlation matrix heatmap
corrplot(correlation_matrix,
         method = "color",
         type = "upper",
         addCoef.col = "black",
         tl.col = "black",
         tl.srt = 45,
         col = colorRampPalette(c("#4477AA", "white", "#EE6677"))(200),
         title = "Correlation Matrix of Growth Rates")
```




```r
ggsave("correlation_heatmap.pdf", width = 10, height = 8)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```r
# A full period correlation plot saved to PDF
pdf("correlation_heatmap_full_period.pdf", width = 12, height = 12)
corrplot(correlation_matrix,
         method = "color",
         type = "full",
         addCoef.col = "black",
         tl.col = "black",
         tl.srt = 45,
         col = colorRampPalette(c("#4477AA", "white", "#EE6677"))(200),
         title = "Correlation Plot (2003-2023)")
dev.off()
```

```
## pdf
##   2
```

```r
# Arrange indicators by absolute correlation with the main luxury growth rate
correlations_with_growth <- correlation_matrix["Growth_Rate", -1]
corr_df <- data.frame(
  Indicator = names(correlations_with_growth),
  Correlation = correlations_with_growth,
  Abs_Correlation = abs(correlations_with_growth)
) %>%
  arrange(desc(Abs_Correlation))

# Bar plot to show the strength of correlation with luxury growth
ggplot(corr_df, aes(x = reorder(Indicator, Abs_Correlation), y = Correlation)) +
  geom_bar(stat = "identity",
           fill = ifelse(corr_df$Correlation > 0, "#4477AA", "#EE6677")) +
  coord_flip() +
  labs(title = "Correlation Strength with Luxury Growth (2003-2023)",
       x = "Indicators",
       y = "Correlation Coefficient") +
  theme_minimal()
```


```r
ggsave("correlation_strength_full_period.pdf", width = 10, height = 8)
```

```r
# Prepare annual mean growth data as a time series
ts_data <- luxury_points %>%
  group_by(Year) %>%
  summarise(Mean_Growth = mean(Growth_Rate, na.rm = TRUE))

ts_growth <- ts(na.omit(ts_data$Mean_Growth),
                start = min(ts_data$Year[!is.na(ts_data$Mean_Growth)]))

print("Time series summary:")
```

```
## [1] "Time series summary:"
```

```r
print(summary(ts_growth))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -9.316   1.573   7.744   7.588  14.380  22.269
```

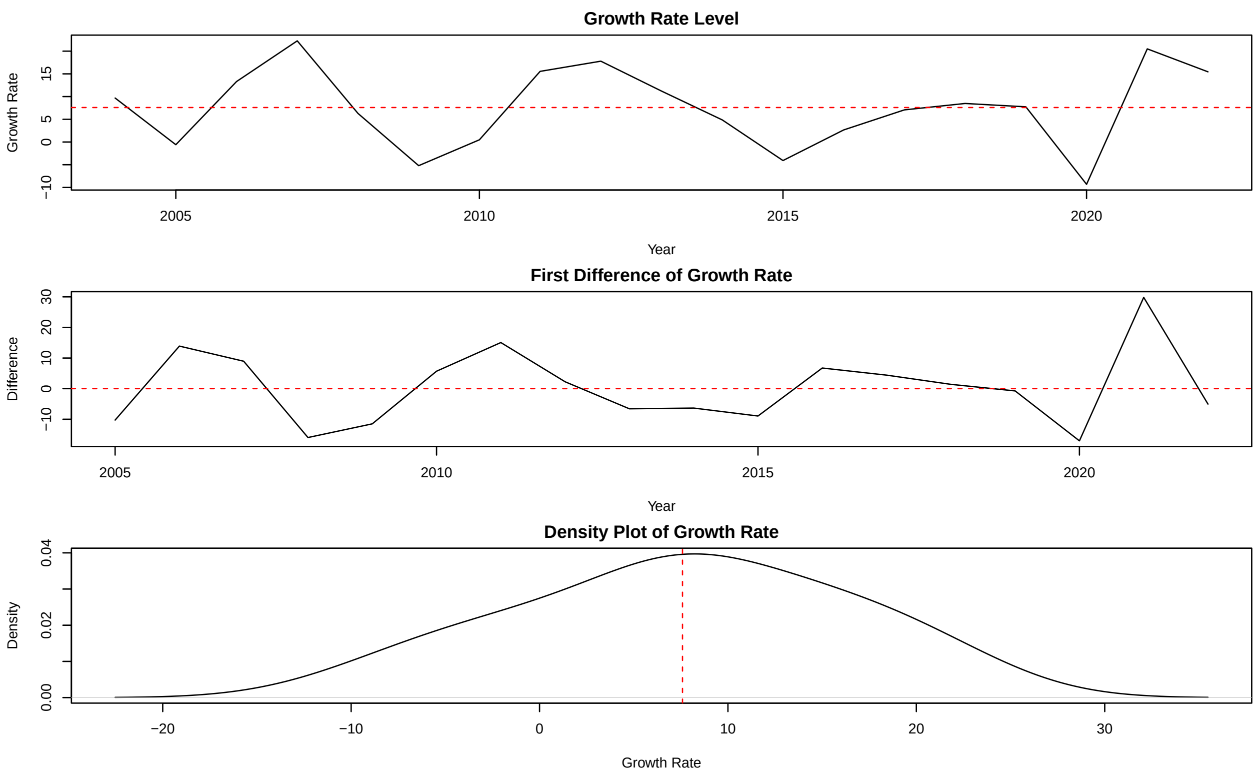
```r
print("Time series range:")
```

```
## [1] "Time series range:"
```

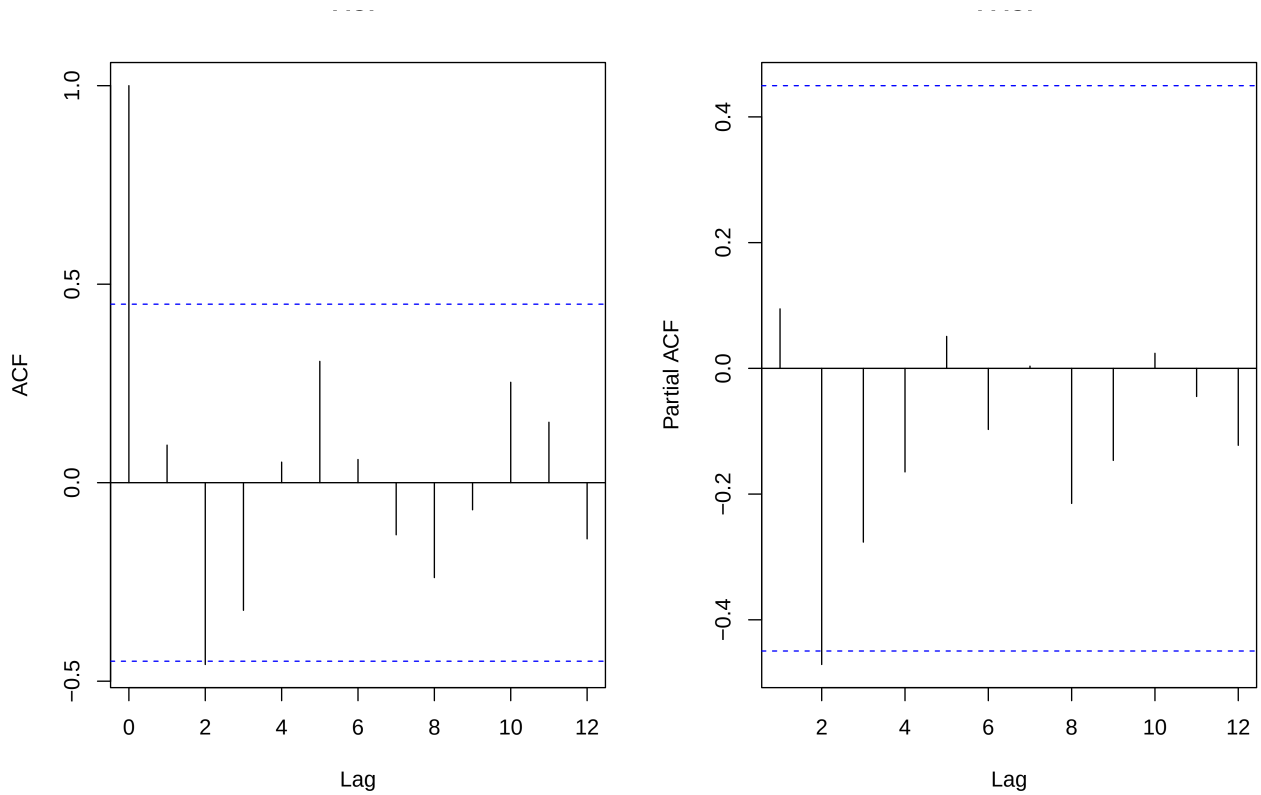
```r
print(range(time(ts_growth)))
```

```
## [1] 2004 2022
```

```r
# Visualize the time series, its first difference, and its density
par(mfrow = c(3,1), mar = c(4,4,2,2))

# Original time series
plot(ts_growth, type = "l", main = "Growth Rate Level",
     xlab = "Year", ylab = "Growth Rate")
abline(h = mean(ts_growth), col = "red", lty = 2)

# First difference to check stationarity
plot(diff(ts_growth), type = "l", main = "First Difference of Growth Rate",
     xlab = "Year", ylab = "Difference")
abline(h = 0, col = "red", lty = 2)

# Density plot of the growth rate distribution
plot(density(ts_growth), main = "Density Plot of Growth Rate",
     xlab = "Growth Rate", ylab = "Density")
abline(v = mean(ts_growth), col = "red", lty = 2)
```



```r
# Autocorrelation and Partial Autocorrelation functions
par(mfrow = c(1,2))
acf(ts_growth, main="ACF")
pacf(ts_growth, main="PACF")
```




```r
# Perform stationarity tests: ADF (Augmented Dickey-Fuller) and KPSS tests
adf_result <- adf.test(ts_growth)
kpss_result <- kpss.test(ts_growth)
```

```
## Warning in kpss.test(ts_growth): p-value greater than printed p-value
```

```r
print("ADF Test Results:")
```

```
## [1] "ADF Test Results:"
```

```r
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_growth
## Dickey-Fuller = -3.6492, Lag order = 2, p-value = 0.04648
## alternative hypothesis: stationary
```

```r
print("KPSS Test Results:")
```

```
## [1] "KPSS Test Results:"
```

```r
print(kpss_result)
```

```
##
##  KPSS Test for Level Stationarity
```

```
##
## data:  ts_growth
## KPSS Level = 0.070184, Truncation lag parameter = 2, p-value = 0.1
```

```r
# Aggregate the indicators at the annual level to create exogenous variables for the ARIMAX model
yearly_indicators <- merged_data %>%
  group_by(Year) %>%
  summarise(
    GDP = mean(GDP.Growth.Rate, na.rm = TRUE),
    Gini = mean(Gini.Change.Rate, na.rm = TRUE),
    Top1 = mean(Top.1..Growth.Rate, na.rm = TRUE),
    Top10 = mean(Top.10..Growth.Rate, na.rm = TRUE),
    Income = mean(Income.Growth.Rate, na.rm = TRUE),
    Bottom50 = mean(Bottom.50..Growth.Rate, na.rm = TRUE),
    Consumer = mean(Consumer.Sentiment.Growth.Rate, na.rm = TRUE),
    .groups = 'drop'
  )

# Create different sets of external regressors (xreg) to test various hypotheses
xreg1 <- cbind(GDP = yearly_indicators$GDP,
               Gini = yearly_indicators$Gini)

xreg2 <- cbind(GDP = yearly_indicators$GDP,
               Gini = yearly_indicators$Gini,
               Top1 = yearly_indicators$Top1,
               Top10 = yearly_indicators$Top10)

xreg3 <- cbind(Income = yearly_indicators$Income,
               Bottom50 = yearly_indicators$Bottom50)

xreg4 <- cbind(GDP = yearly_indicators$GDP,
               Consumer = yearly_indicators$Consumer)

xreg5 <- cbind(Top1 = yearly_indicators$Top1,
               Bottom50 = yearly_indicators$Bottom50,
               Gini = yearly_indicators$Gini)

xreg6 <- cbind(GDP = yearly_indicators$GDP,
               Gini = yearly_indicators$Gini,
               Top1 = yearly_indicators$Top1)
```

```r
# Fit ARIMAX models using different sets of regressors
# ARIMA order chosen as (2,1,1) here as a starting point
models <- list(
    model1 = Arima(ts_growth, order=c(2,1,1), xreg=xreg1),
    model2 = Arima(ts_growth, order=c(2,1,1), xreg=xreg2),
    model3 = Arima(ts_growth, order=c(2,1,1), xreg=xreg3),
    model4 = Arima(ts_growth, order=c(2,1,1), xreg=xreg4),
    model5 = Arima(ts_growth, order=c(2,1,1), xreg=xreg5),
    model6 = Arima(ts_growth, order=c(2,1,1), xreg=xreg6)
)

# Compare models by AIC and BIC criteria
results <- data.frame(
    Model = c("Strong_Correlation(GDP+Gini)",
```

```r
              "All_Positive(GDP+Gini+Top1+Top10)",
              "Negative(Income+Bottom50)",
              "GDP_Led(GDP+Consumer)",
              "Inequality(Top1+Bottom50+Gini)",
              "Comprehensive(GDP+Gini+Top1)"),
    AIC = sapply(models, AIC),
    BIC = sapply(models, BIC)
)

print("Model Comparison Results (sorted by AIC):")
```

```
## [1] "Model Comparison Results (sorted by AIC):"
```

```r
print(results[order(results$AIC),])
```

```
##                                    Model      AIC      BIC
## model1      Strong_Correlation(GDP+Gini) 127.7397 133.0819
## model4             GDP_Led(GDP+Consumer) 128.7584 134.1006
## model2 All_Positive(GDP+Gini+Top1+Top10) 129.0709 136.1938
## model6      Comprehensive(GDP+Gini+Top1) 129.3231 135.5557
## model5    Inequality(Top1+Bottom50+Gini) 136.2716 142.5042
## model3         Negative(Income+Bottom50) 139.0217 144.3639
```

```r
# Evaluate model accuracy for each fitted model
accuracy_results <- do.call(rbind, lapply(models, accuracy))
rownames(accuracy_results) <- paste0("Model", 1:6)

print("\nAccuracy Comparison:")
```

```
## [1] "\nAccuracy Comparison:"
```

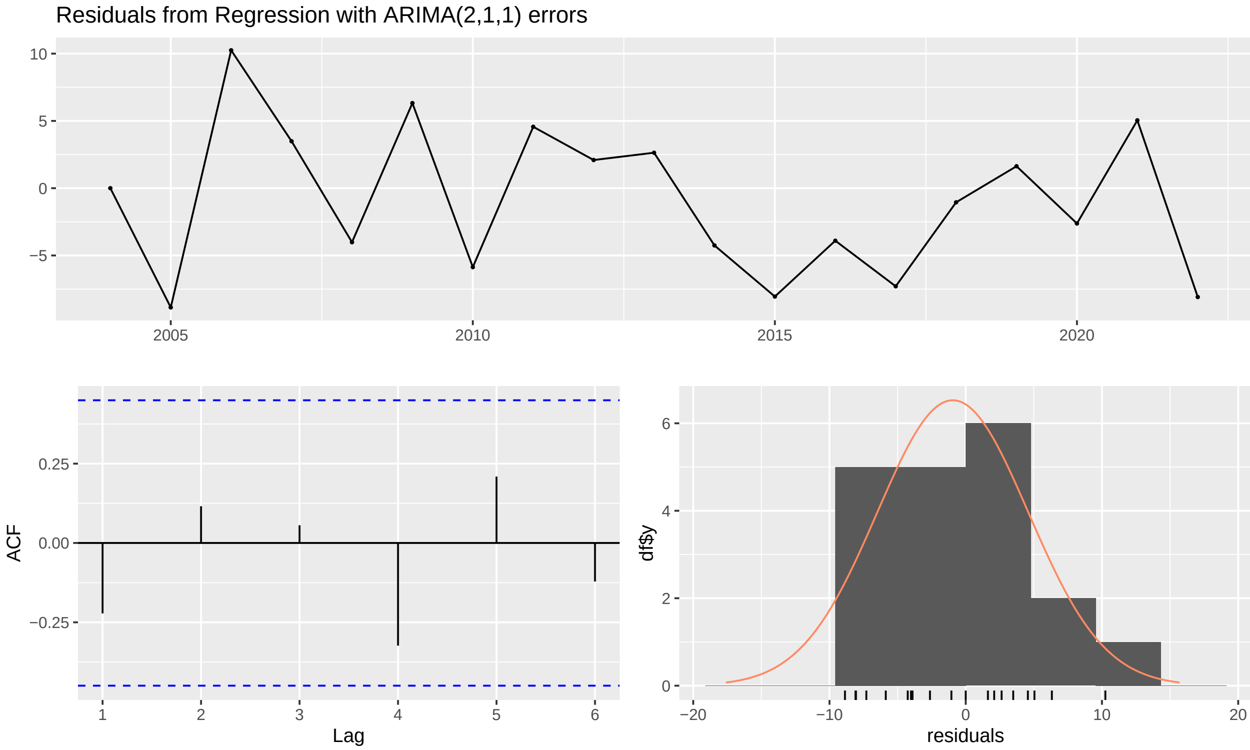
```r
print(accuracy_results)
```

```
##                ME     RMSE      MAE        MPE     MAPE      MASE       ACF1
## Model1 -0.9468362 5.484167 4.739250   6.988534 195.5292 0.4996203 -0.2218850
## Model2 -0.6830317 4.763722 4.019023 -30.025873 189.7811 0.4236927 -0.2859690
## Model3 -0.9963809 7.158906 5.820465   4.142356 211.4044 0.6136040 -0.1746484
## Model4 -0.5205585 5.684369 4.482330   7.710616 164.9878 0.4725354 -0.1747505
## Model5 -0.6507357 6.183542 5.145978  -5.743029 233.0814 0.5424983 -0.1785380
## Model6 -0.9555528 5.455636 4.435546  24.559221 170.5195 0.4676032 -0.1793493
```

```r
# Identify the best model by AIC
best_model_index <- which.min(results$AIC)
cat("\nDiagnostic Tests for Best Model (AIC):\n")
```

```
##
## Diagnostic Tests for Best Model (AIC):
```

```r
# Check residuals of the best model (diagnostic plots and tests)
checkresiduals(models[[best_model_index]])
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,1,1) errors
## Q* = 5.9671, df = 3, p-value = 0.1132
##
## Model df: 3.   Total lags used: 6
```

```
# We can try various ARIMA orders and test model diagnostics systematically

arima_orders <- list(
    c(1,1,1), c(2,1,1), c(2,1,2),
    c(3,1,1), c(3,1,2), c(3,1,3),
    c(2,1,3), c(1,1,2)
)

test_models <- function(xreg_data) {
    results <- data.frame()
    for(order in arima_orders) {
        tryCatch({
            model <- Arima(ts_growth, order=order, xreg=xreg_data)

            # Conduct residual diagnostics
            lb_test <- Box.test(residuals(model), lag=10, type="Ljung-Box")
            sw_test <- shapiro.test(residuals(model))
            res <- residuals(model)

            # Breusch-Pagan test for heteroskedasticity
            bp_data <- data.frame(
                res = res^2,
```

```r
                xreg_data
            )
            bp_model <- lm(res ~ ., data=bp_data)
            bp_test <- bptest(bp_model)

            # Consider a model "valid" if residuals pass these tests:
            # 1. No autocorrelation (Ljung-Box p > 0.05)
            # 2. Residuals are normally distributed (Shapiro-Wilk p > 0.05)
            # 3. No heteroskedasticity (Breusch-Pagan p > 0.05)
            if(lb_test$p.value > 0.05 &&
               sw_test$p.value > 0.05 &&
               bp_test$p.value > 0.05) {

                temp <- data.frame(
                    AR = order[1],
                    I = order[2],
                    MA = order[3],
                    AIC = AIC(model),
                    BIC = BIC(model),
                    LB_pvalue = lb_test$p.value,
                    SW_pvalue = sw_test$p.value,
                    BP_pvalue = bp_test$p.value,
                    RMSE = sqrt(mean(res^2)),
                    MAE = mean(abs(res))
                )
                results <- rbind(results, temp)
            }
        }, error = function(e) {
            cat("Error with order", order, ":", conditionMessage(e), "\n")
        })
    }
    return(results)
}

# Test refinements on the model with xreg2
model2_results <- test_models(xreg2)

if(nrow(model2_results) > 0) {
    cat("\nValid Models (passing all diagnostic tests):\n")
    print(model2_results[order(model2_results$AIC),])

    best_order_index <- which.min(model2_results$AIC)
    best_order <- c(
        model2_results$AR[best_order_index],
        model2_results$I[best_order_index],
        model2_results$MA[best_order_index]
    )

    cat("\nBest ARIMA order (among valid models):",
        paste("AR=", best_order[1],
              ", I=", best_order[2],
              ", MA=", best_order[3]))
```

```r
    cat("\n\nDiagnostic p-values for best model:")
    cat("\nLjung-Box p-value:", model2_results$LB_pvalue[best_order_index])
    cat("\nShapiro-Wilk p-value:", model2_results$SW_pvalue[best_order_index])
    cat("\nBreusch-Pagan p-value:", model2_results$BP_pvalue[best_order_index])
} else {
    cat("\nNo models passed all diagnostic tests. Consider:")
    cat("\n1. Different ARIMA orders")
    cat("\n2. Data transformations")
    cat("\n3. Different variable combinations")
}
```

```
##
## Valid Models (passing all diagnostic tests):
##     AR I MA      AIC      BIC LB_pvalue SW_pvalue BP_pvalue     RMSE      MAE
## BP   2 1  1 129.0709 136.1938 0.2354933 0.4904064 0.1349317 4.763722 4.019023
## BP3  1 1  2 132.1000 139.2229 0.4550168 0.1234190 0.1548043 5.214337 3.983712
## BP1  2 1  2 132.6601 140.6735 0.6526678 0.6456542 0.3234843 4.973134 4.160845
## BP2  2 1  3 133.0016 141.9053 0.7262155 0.1651825 0.7005599 4.323447 3.646410
##
## Best ARIMA order (among valid models): AR= 2 , I= 1 , MA= 1
##
## Diagnostic p-values for best model:
## Ljung-Box p-value: 0.2354933
## Shapiro-Wilk p-value: 0.4904064
## Breusch-Pagan p-value: 0.1349317
```

```r
final_model <- Arima(ts_growth,
                     order=c(2,1,1),
                     xreg=xreg2)
```

```r
ts_growth_2022 <- window(ts_growth, end=2022)
xreg2_2022 <- xreg2[yearly_indicators$Year <= 2022, ]

final_model_2022 <- Arima(ts_growth_2022,
                          order=c(2,1,1),
                          xreg=xreg2_2022)

future_xreg_2023 <- matrix(c(
    6.59,     # 2023 GDP Growth Rate (example)
    0,        # 2023 Gini Change Rate (example)
    0.0574,   # 2023 Top1 Growth Rate (example)
    0.0283    # 2023 Top10 Growth Rate (example)
), nrow=1)

forecast_2023 <- forecast(final_model_2022,
                          xreg=future_xreg_2023,
                          h=1)
```

```
## Warning in forecast.forecast_ARIMA(final_model_2022, xreg = future_xreg_2023, :
## xreg contains different column names from the xreg used in training. Please
## check that the regressors are in the same order.
```

```r
cat("\nForecast Results for 2023:\n")
```

```
##
## Forecast Results for 2023:
```

```r
cat("Point Forecast:", round(forecast_2023$mean, 2), "\n")
```

```
## Point Forecast: 8.15
```

```r
cat("95% Confidence Interval:",
    round(forecast_2023$lower[,"95%"], 2), "to",
    round(forecast_2023$upper[,"95%"], 2), "\n")
```

```
## 95% Confidence Interval: -4.47 to 20.76
```

```r
actual_2023 <- 9

cat("\nComparison with Actual 2023 Value:\n")
```

```
##
## Comparison with Actual 2023 Value:
```

```r
cat("Predicted:", round(forecast_2023$mean, 2), "\n")
```

```
## Predicted: 8.15
```

```r
cat("Actual:", round(actual_2023, 2), "\n")
```

```
## Actual: 9
```

```r
cat("Difference:", round(actual_2023 - forecast_2023$mean, 2), "\n")
```

```
## Difference: 0.85
```

```r
plot(forecast_2023,
     main="Luxury Market Growth Rate Forecast for 2023",
     ylab="Growth Rate (%)",
     xlab="Year")

points(2023, actual_2023, pch=19, col="red")
legend("topright",
       legend=c("Actual", "Forecast"),
       col=c("red", "blue"),
       pch=c(19, 19))
```


```r
prediction_error <- actual_2023 - forecast_2023$mean
cat("\nPrediction Accuracy Metrics:\n")
```

```
##
## Prediction Accuracy Metrics:
```

```r
cat("Absolute Error:", round(abs(prediction_error), 2), "\n")
```

```
## Absolute Error: 0.85
```

```r
cat("Percentage Error:", round(abs(prediction_error/actual_2023)*100, 2), "%\n")
```

```
## Percentage Error: 9.49 %
```