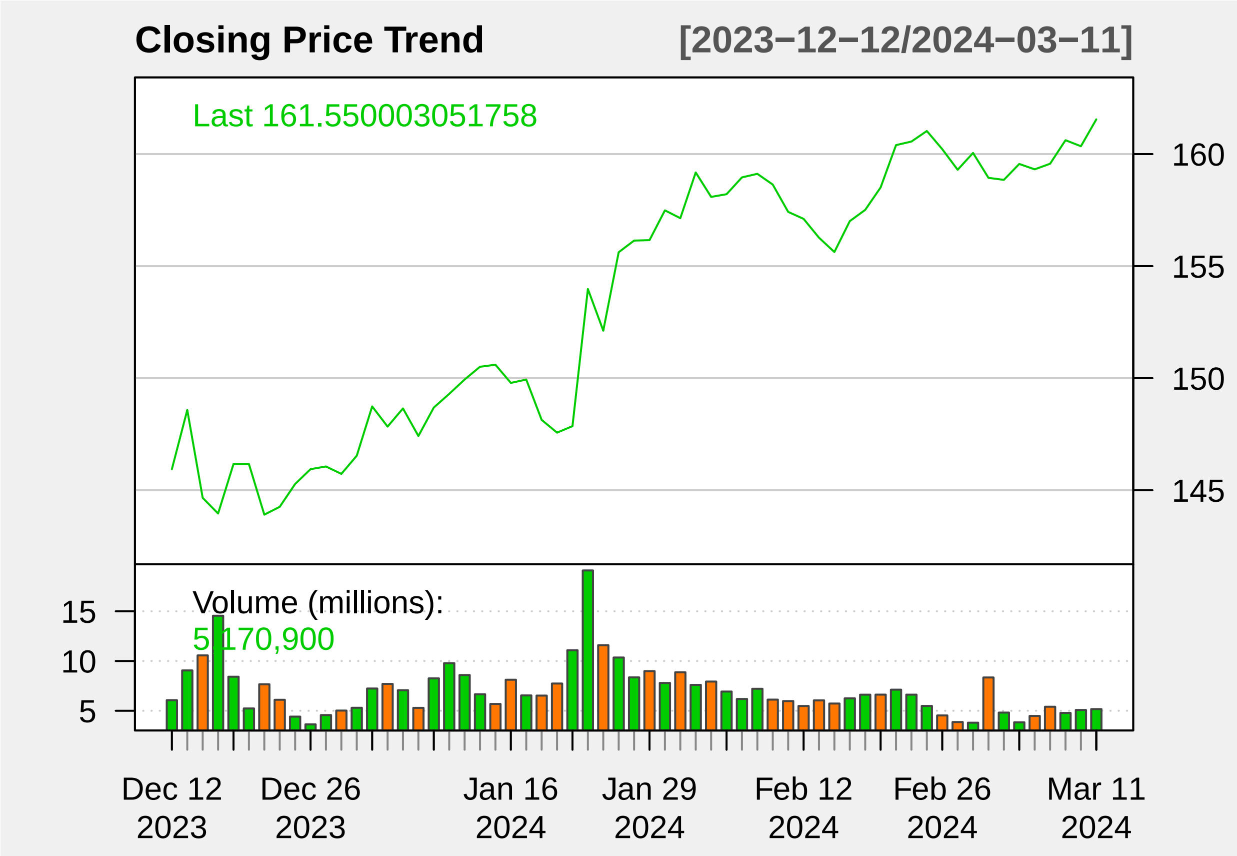
PG

2024-03-19

```r
# Install and load the quantmod package
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
# Specify the stock symbol and the time period
stock_symbol <- "PG"
start_date <- "2023-12-12"
end_date <- "2024-03-12"

# Fetch the stock data
PG_cp_trend <- getSymbols(stock_symbol, from = start_date, to = end_date)

# Plot the closing price trend
chartSeries(PG, type = "line", theme = "white", name = "Closing Price Trend")
```




```r
# Read the CSV file
PG <- read.csv("P_and_G_StockInfo.csv")
```

Data Cleaning

PG data cleaning

```r
PG$Date <- as.Date(PG$Date, format = "%m/%d/%y")
# Sort the data by the date column
PG <- PG[order(PG$Date), ]

PG <- na.omit(PG)
```

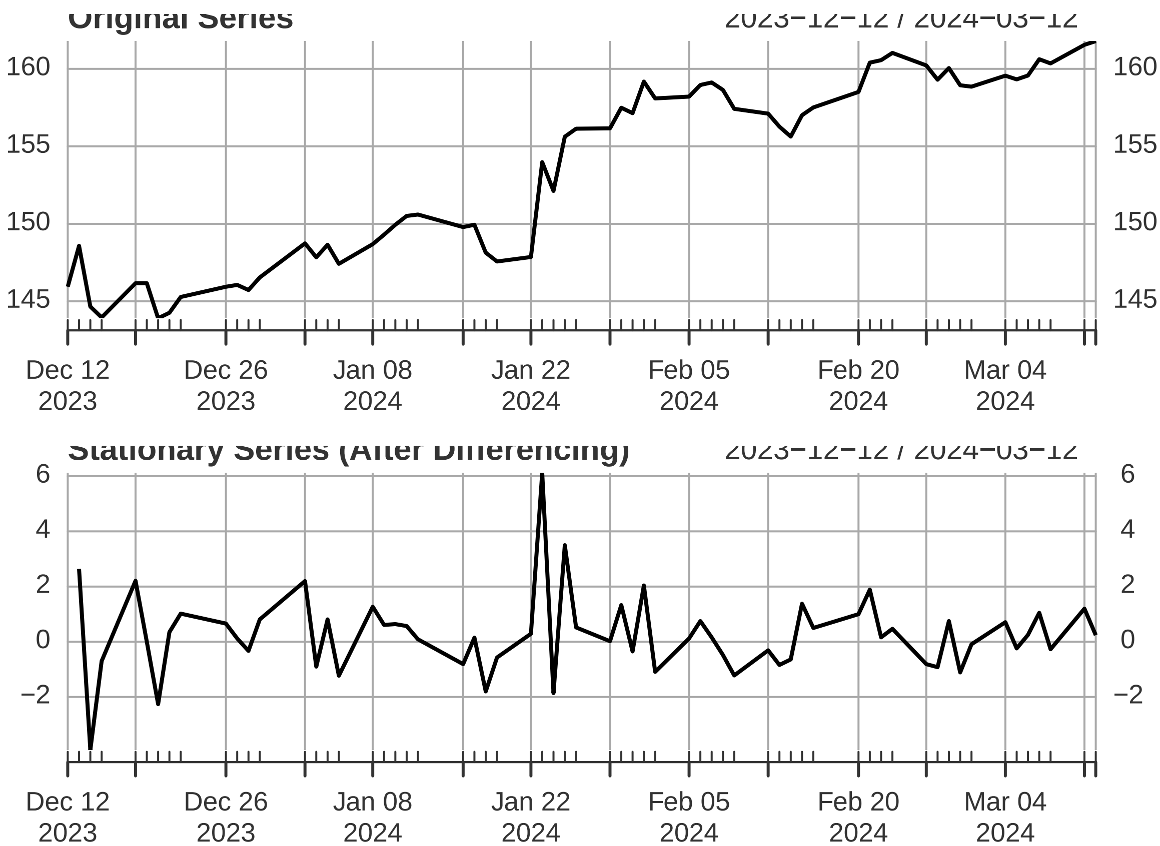
Training

```r
train_PG <- subset(PG, Date >= as.Date("2023-12-12") & Date <= as.Date("2024-02-12"))
test_PG <- subset(PG, !(Date >= as.Date("2023-12-12") & Date <= as.Date("2024-02-12")))
closing_prices <- xts(PG$Close, order.by = PG$Date)
library(tseries)

# Perform the Augmented Dickey-Fuller test
```

```r
adf_result <- adf.test(closing_prices)

# Print the test results
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  closing_prices
## Dickey-Fuller = -2.4328, Lag order = 3, p-value = 0.3997
## alternative hypothesis: stationary
```

```r
stationary_series <- diff(closing_prices)

# Plot the original and differenced series
par(mfrow=c(2,1))
plot(closing_prices, main="Original Series", type='l')
plot(stationary_series, main="Stationary Series (After Differencing)", type='l')
```




```r
stationary_series <- na.omit(stationary_series)

adf_result_diff <- adf.test(stationary_series)

# Print the test results
print(adf_result_diff)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  stationary_series
## Dickey-Fuller = -3.5713, Lag order = 3, p-value = 0.04313
## alternative hypothesis: stationary
```

```
CP_train<- xts(train_PG$Close, order.by = train_PG$Date)
CP_test <- xts(test_PG$Close, order.by = test_PG$Date)
library(forecast)

start_date <- as.Date(start_date)
time_series <- ts(data = CP_train, start = start_date, frequency = 1)

# Identify the best ARIMA model using auto.arima
arima_model <- auto.arima(time_series)

# Print the identified ARIMA model
print(arima_model)
```

```
## Series: time_series
## ARIMA(0,1,0)
##
## sigma^2 = 2.761:  log likelihood = -78.99
## AIC=159.98   AICc=160.08   BIC=161.69
```

```
arima_model_1 <- arima(CP_train, order = c(0,1,0))
arima_model_2 <- arima(CP_train, order = c(2,1,2))
summary(arima_model_1)
```

```
##
## Call:
## arima(x = CP_train, order = c(0, 1, 0))
##
##
## sigma^2 estimated as 2.76:  log likelihood = -78.99,  aic = 159.98
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE      MAPE      MASE
## Training set 0.2694271 1.641605 1.118475 0.1720142 0.7428682 0.9792327
##                   ACF1
## Training set -0.2735432
```

```
summary(arima_model_2)
```

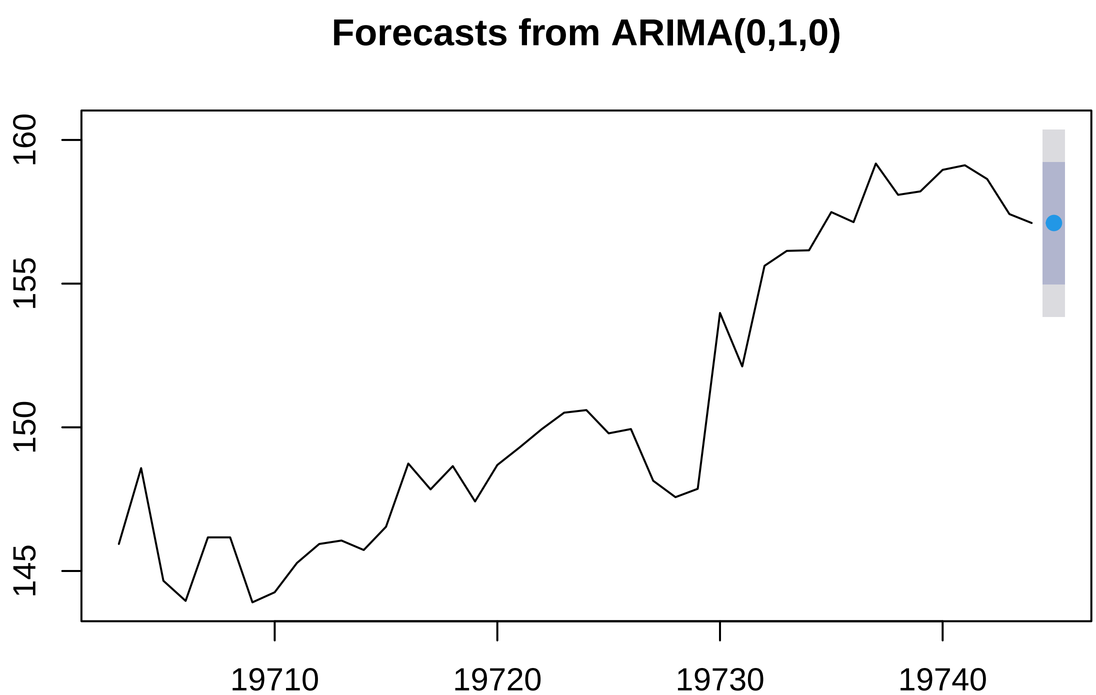
```
##
## Call:
## arima(x = CP_train, order = c(2, 1, 2))
##
## Coefficients:
##          ar1      ar2      ma1     ma2
##       0.4580  -0.7674  -0.7075  0.9418
```

```
## s.e.  0.1556   0.2233   0.1363  0.2598
##
## sigma^2 estimated as 2.332:  log likelihood = -76.36,  aic = 162.72
##
## Training set error measures:
##                     ME     RMSE      MAE      MPE      MAPE      MASE
## Training set 0.2726283 1.509021 1.048456 0.174925 0.6934678 0.9179307
##                     ACF1
## Training set -0.08501235
```

```r
test_PG$date <- as.Date(test_PG$Date)

# Perform one-step forecast without re-estimation
one_step_forecast <- forecast(arima_model, h = 1, newdata = test_PG)
```

```
## Warning in forecast.forecast_ARIMA(arima_model, h = 1, newdata = test_PG): The
## non-existent newdata arguments will be ignored.
```

```r
# plot the one-step forecast
plot(one_step_forecast)

points(test_PG$Date, test_PG$Close, col = 'red', type = 'p')
```



```
forecast_length <- nrow(test_PG)  # Number of rows to forecast
forecasted_values <- forecast(arima_model, h = forecast_length+50)

# Plot the forecast with historical data
plot(forecasted_values)

# Overlay the actual data points on the forecast plot
lines(test_PG$Date, test_PG$Close, col = 'red')
```