Indirect Cost Forecasting

Md Ismail Hossain

2025-01-30

R Markdown

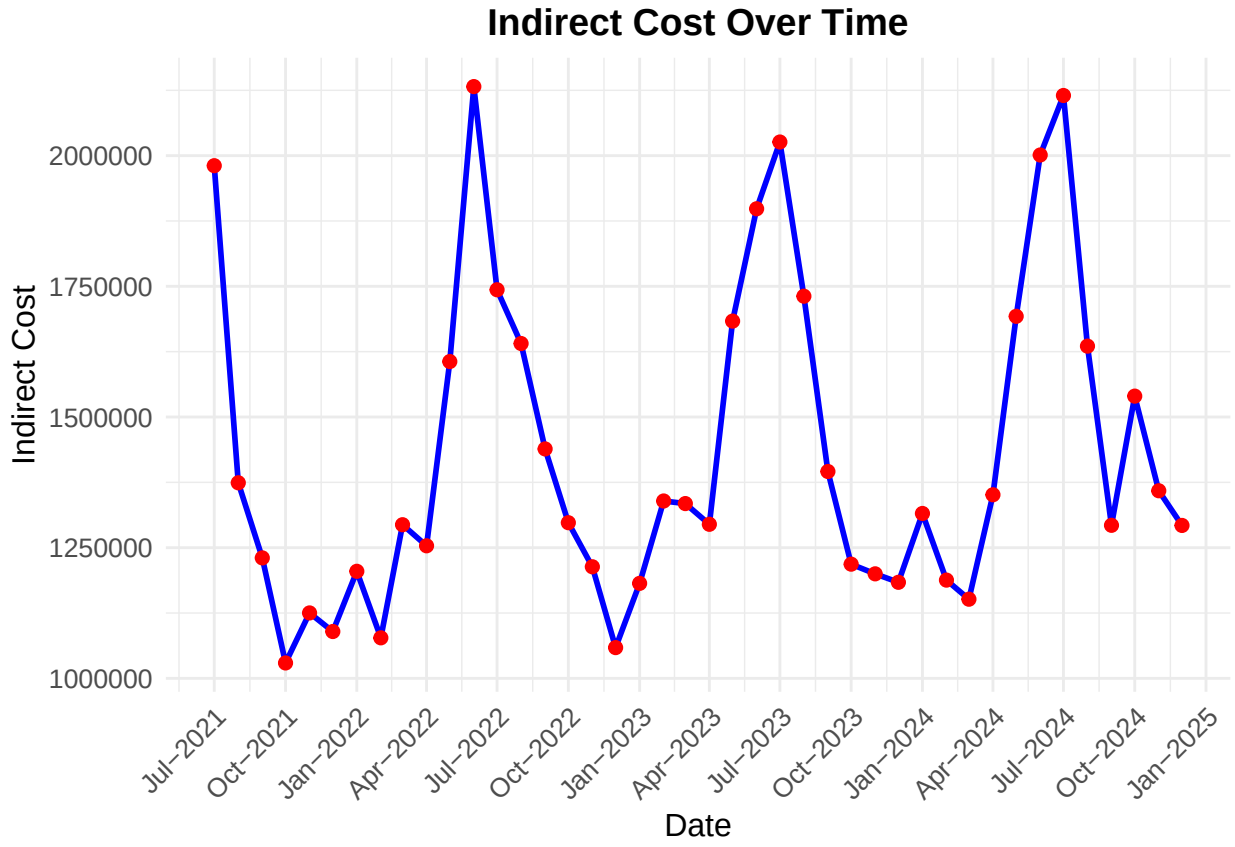
```
# Load necessary libraries
library(forecast)
```

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

```
library(tseries)
library(readr)
library(readxl)
library(ggplot2)


# Read the data
data <- read_excel("C:/Users/mhossa11/OneDrive - University of Wyoming/Projects/Indirect Cost Forecasti
    sheet = "Clean Data")

# Convert Date column to Date format
data$Date <- as.Date(data$Date, format="%m/%d/%Y")

# Create a time series object
ts_data <- ts(data$Indirect_Cost, start=c(2021,7), frequency=12)  # Monthly data starting from July 202

# Create ggplot visualization
ggplot(data, aes(x = Date, y = Indirect_Cost)) +
  geom_line(color = "blue", size = 1) +  # Line color and thickness
  geom_point(color = "red", size = 2) +  # Points for better visibility
  scale_x_date(date_breaks = "3 months", date_labels = "%b-%Y") +  # Show x-axis labels for every 3 mon
  labs(title = "Indirect Cost Over Time",
       x = "Date",
       y = "Indirect Cost") +
  theme_minimal() +  # Apply a minimal clean theme
  theme(axis.text.x = element_text(angle = 45, hjust = 1, size = 10),  # Rotate x-axis labels
        plot.title = element_text(hjust = 0.5, face = "bold", size = 14),
        axis.title = element_text(size = 12),
        axis.text = element_text(size = 10))
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
```

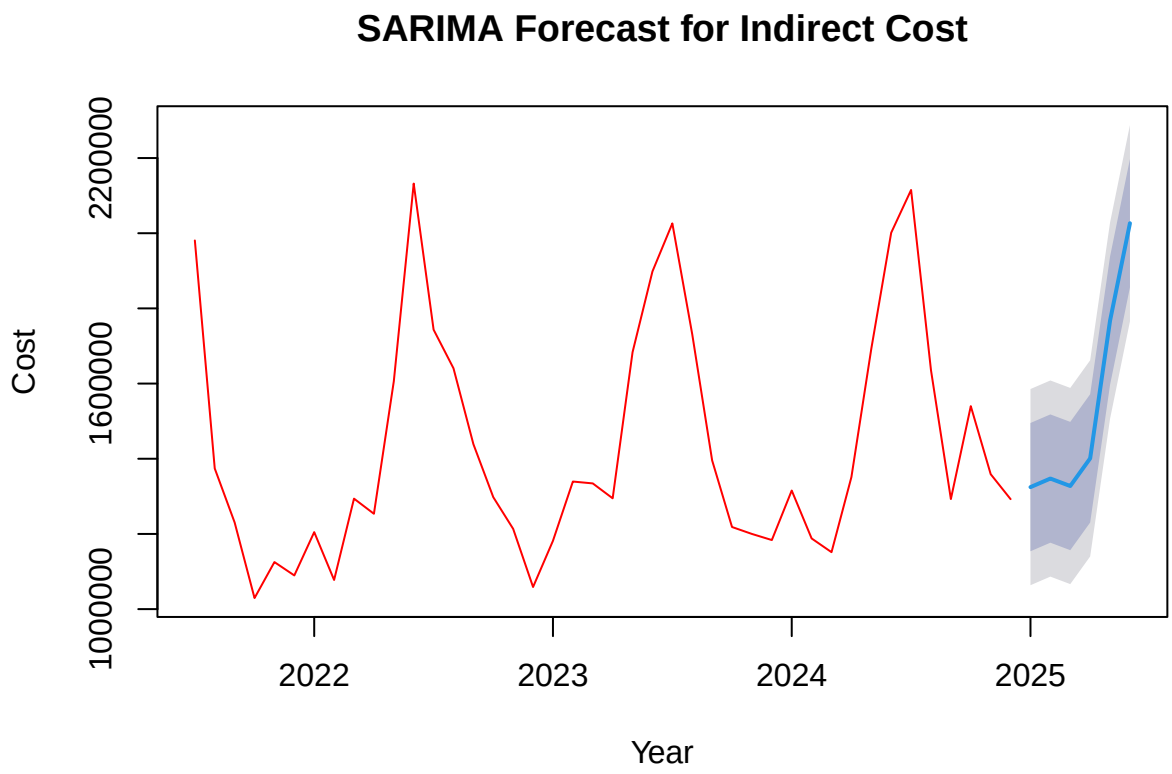
```
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```




```r
# Check stationarity using Augmented Dickey-Fuller (ADF) test
adf_test <- adf.test(ts_data)

# Differencing if needed
if(adf_test$p.value > 0.05) {
  diff_data <- diff(ts_data, differences=1)  # First differencing
  adf_test_diff <- adf.test(diff_data)

  # Seasonal differencing if required
  if(adf_test_diff$p.value > 0.05) {
    diff_data <- diff(diff_data, lag=12)  # Seasonal differencing
  }
} else {
  diff_data <- ts_data  # Already stationary
}

# Auto ARIMA to find best SARIMA parameters
best_model <- auto.arima(ts_data, seasonal=TRUE)

# Print model summary
summary(best_model)
```

```
## Series: ts_data
## ARIMA(0,0,0)(1,1,0)[12] with drift
##
## Coefficients:
##          sar1     drift
##       -0.5272  4342.358
## s.e.   0.1834  1506.108
##
## sigma^2 = 1.772e+10:  log likelihood = -397.46
## AIC=800.91   AICc=801.84   BIC=805.12
##
## Training set error measures:
##                     ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 1297.762 108693.3 73537.75 -0.04184224 5.003099 0.5620852
##                      ACF1
## Training set -0.04643894
```

```
# Forecast for next 12 months
future_forecast <- forecast(best_model, h=6)

# Plot the forecast
plot(future_forecast, main="SARIMA Forecast for Indirect Cost", ylab="Cost", xlab="Year", col="red")
```



```r
# Print forecast values
print(future_forecast)
```

```
##          Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## Jan 2025        1324539 1153937 1495141 1063625 1585452
## Feb 2025        1347334 1176732 1517936 1086421 1608248
## Mar 2025        1327454 1156852 1498056 1066540 1588367
## Apr 2025        1401024 1230421 1571626 1140110 1661937
## May 2025        1767380 1596778 1937982 1506467 2028294
## Jun 2025        2026608 1856006 2197210 1765694 2287521
```