Project 1 | Econ 684: Time Series Forecasting

Cassia Rodrigues- UID:119129180

2024-04-04

setwd("/Users/User/Desktop/Applied Economics/ECON 684/Project 1")

"'{r:ARIMA}

arima_model <- auto.arima(chicago_ts) # Forecasting for the next 24 months (2 years) future_forecast <- forecast(arima_model, h = 24)

Plot the forecast with the year 2024 on the y-axis

plot(future_forecast, main = "Forecasted Rental Price Index")

Add a label for the year 2024 on the y-axis

mtext("2024", side = 2, line = 2, col = "red")

plot(future_forecast, main = "Forecasted Rental Price Index") print(future_forecast) plot(chicago_ts, main = "Original vs. Forecasted Rental Price Index") lines(future_forecast$mean, col = "red")

Print summary of the selected ARIMA model

print(summary(arima_model))

Extract model parameters

order <- arima_model$arma$seasonal_order <- $arima_model$arma[4:6]

```{r: Decomposition}
library(dplyr)

# 'chicago_ts' is your monthly time series data
plot(chicago_ts, main = "Average Monthly Rental Prices Over Time", xlab = "Month", ylab = "Rental Price

decomposition <- decompose(chicago_ts)

plot(decomposition)

# Extract each component from the decomposition object
trend_component <- decomposition$trend
seasonal_component <- decomposition$seasonal
random_component <- decomposition$random

# Plot each component separately
plot(chicago_ts, main = "Original Time Series", xlab = "Month", ylab = "Rental Price")
lines(trend_component, col = "blue", lwd = 2)  # Add trend component to the plot
legend("bottomright", legend = c("Original", "Trend"), col = c("black", "blue"), lty = 1, lwd = c(1, 2))

# Add vertical line at year 2020
abline(v = as.Date("2020-01-01"), col = "red", lty = 2)

plot(seasonal_component, main = "Seasonal Component", xlab = "Month", ylab = "Rental Price")
plot(random_component, main = "Random (Residual) Component", xlab = "Month", ylab = "Rental Price")
```

"'{r: Seasonal Adjustment}

library(stats)

decomposition <- decompose(chicago_ts) seasonal_component <- decomposition$seasonal seasonally_adjusted_rental_index <- chicago_ts - seasonal_component

plot(seasonally_adjusted_rental_index, main = "Seasonally Adjusted Rental Price Index", xlab = "Month", ylab = "Seasonally Adjusted Rental Price")

```{r: Exponential Smoothing}
library(forecast)

# Fit exponential smoothing model
ets_model <- ets(chicago_ts)

# Forecast future rental prices
ets_forecast <- forecast(ets_model, h = 24)  # Forecasting for the next 24 months (2 years)

# Plot the forecast
```

```
  plot(ets_forecast, main = "Exponential Smoothing Forecast for Rental Price Index", xlab = "Month", yla

# Calculate forecast accuracy metrics
accuracy_metrics <- accuracy(ets_forecast)

# Print the accuracy metrics
print(accuracy_metrics)


library(ggplot2)

# Extract dates from forecast
forecast_dates <- seq(as.Date("2022-01-01"), by = "month", length.out = length(ets_forecast$mean))

# Create data frame with forecasted values and dates
ets_forecast_df <- data.frame(date = forecast_dates,
                              forecast = ets_forecast$mean,
                              lower = ets_forecast$lower[, "95%"],
                              upper = ets_forecast$upper[, "95%"])

# Plot forecast using ggplot2
ggplot(ets_forecast_df, aes(x = date)) +
  geom_line(aes(y = forecast), color = "blue") +
  geom_ribbon(aes(ymin = lower, ymax = upper), fill = "lightblue", alpha = 0.5) +
  geom_vline(xintercept = as.numeric(as.Date("2020-01-01")), linetype = "dashed", color = "red") +
  labs(title = "Exponential Smoothing Forecast for Rental Price Index",
       x = "Date", y = "Rental Price") +
  theme_minimal() +
  theme(panel.grid.major = element_line(color = "gray", linetype = "dashed"))
```