# Time Series Forecasting:

# ETS vs ARIMA vs Regression

# Using R script

1. **Data Exploration**

# Load necessary libraries

library(tsutils) # For time series utility functions

library(forecast) # For forecasting functions

library(tseries) # For time series analysis functions

library(readr) # For reading CSV files

# Read data files

d1982 <- read\_csv("1982.csv") # Read the data for the year 1982

d1983 <- read\_csv("1983.csv") # Read the data for the year 1983

# Convert to time series objects

d1982\_ts <- ts(d1982, frequency = 12, start = c(1982,1))

d1983\_ts <- ts(d1983, frequency = 12, start = c(1983,1))

# Centred Moving Average

d1982\_cma <- cmav(d1982\_ts, ma = 12, fill = FALSE) # Compute the centered moving average for 1982

d1983\_cma <- cmav(d1983\_ts, ma = 12, fill = FALSE) # Compute the centered moving average for 1983

# Plot for 1982 data

plot(d1982\_ts, main = "Time Series Data 1982 with CMA", ylab = "Values", xlab = "Years", col = "black")

lines(d1982\_cma, col = "red", lty = 2)

legend("topright", legend = c("Original", "CMA"), col = c("black", "red"), lty = c(1, 2))

# Plot the original time series data

plot(d1983\_ts, main = "Time Series Data 1983 with CMA", ylab = "Values", xlab = "Years", col = "black")

# Add the CMA to the plot, ensuring it starts from 1983

lines(d1983\_cma, col = "red", lty = 2)

# Add a legend to the plot

legend("topright", legend = c("Original", "CMA"), col = c("black", "red"), lty = c(1, 2))

# Seasonal Plots

par(mfrow = c(1, 2)) # Set up a 1x2 plotting area

seasplot(d1982\_ts, main = "Seasonal Plot 1982") # Seasonal plot for 1982

seasplot(d1983\_ts, main = "Seasonal Plot 1983") # Seasonal plot for 1983

par(mfrow = c(1, 1)) # Reset plotting area to default

# Decomposition of time series

decomposition1982 <- decomp(d1982\_ts, decomposition = "additive", outplot = TRUE) # Decompose 1982 time series using multiplicative model

decomposition1983 <- decomp(d1983\_ts, decomposition = "additive", outplot = TRUE) # Decompose 1983 time series using multiplicative model

# Extract the Trend, Seasonality, and Irregular components

trend1982 <- decomposition1982$trend # Extract trend component for 1982

season1982 <- decomposition1982$season # Extract seasonal component for 1982

irregular1982 <- decomposition1982$irregular # Extract irregular component for 1982

trend1983 <- decomposition1983$trend # Extract trend component for 1983

season1983 <- decomposition1983$season # Extract seasonal component for 1983

irregular1983 <- decomposition1983$irregular # Extract irregular component for 1983

# Multiply the Trend and Seasonality for regular components

regular\_components1982 <- trend1982 \* season1982 # Compute regular components for 1982

regular\_components1983 <- trend1983 \* season1983 # Compute regular components for 1983

# Compute the errors

mmm\_errors1982 <- d1982\_ts - regular\_components1982 # Compute errors for 1982

mmm\_errors1983 <- d1983\_ts - regular\_components1983 # Compute errors for 1983

# Compare irregular components with errors

decomposition1982$irregular - mmm\_errors1982 # Compare irregular component and errors for 1982

decomposition1983$irregular - mmm\_errors1983 # Compare irregular component and errors for 1983

# Stationarity Test (ADF Test)

adf\_test\_1982 <- adf.test(d1982\_ts) # Perform ADF test for 1982

adf\_test\_1983 <- adf.test(d1983\_ts) # Perform ADF test for 1983

# Print ADF test results

print(adf\_test\_1982) # Print results for 1982

print(adf\_test\_1983) # Print results for 1983

# ACF and PACF plots

par(mfrow = c(2, 2)) # Set up a 2x2 plotting area

acf(d1982\_ts, main = "ACF of 1982 Time Series") # ACF plot for 1982

pacf(d1982\_ts, main = "PACF of 1982 Time Series") # PACF plot for 1982

acf(d1983\_ts, main = "ACF of 1983 Time Series") # ACF plot for 1983

pacf(d1983\_ts, main = "PACF of 1983 Time Series") # PACF plot for 1983

par(mfrow = c(1, 1)) # Reset plotting area to default

1. **ETS Model**

# Load necessary libraries

library(tsutils) # For time series utility functions

library(forecast) # For forecasting functions

library(tseries) # For time series analysis functions

library(readr) # For reading CSV files

library(smooth) # For smoothing

# Read data files 1982

d1982 <- read\_csv("1982.csv") # Read the data for the year 1982

# Convert to time series objects

d1982\_ts <- ts(d1982, frequency = 12, start = c(1982,1)) # Convert 1982 data to a time series object with monthly frequency

# Forecast Model Builiding

# Find the total number of observations

d1982length <- length(d1982\_ts) # Get the number of observations in the 1982 time series

# Write down size of train set

d1982train\_length <- 130 # Define the size of the training set (first 130 observations)

# And the forecasting horizon

d1982h <- 14 # Define the forecasting horizon (14 steps ahead)

# Create the training set

d1982train <- ts(d1982\_ts[1:d1982train\_length], frequency = 12, start = c(1982, 1))

# Create the test set

d1982test <- ts(d1982\_ts[(d1982train\_length + 1):d1982length], frequency = 12, start = c(1982, 1))

# CHECK ERROR

# Fit an ETS model with additive error, trend, and seasonality components

# using the 'ANN' (Additive error, no trend, no seasonality) model

# using the 'MNN' (Multiplicative error, no trend, no seasonality) model

d1982ETS\_ANN <- ets(d1982train, model = "ANN")

d1982ETS\_MNN <- ets(d1982train, model = "MNN")

summary(d1982ETS\_ANN)

summary(d1982ETS\_MNN)

# Generate the forecast using the ETS model for 14 steps ahead

d1982ETS\_MNN\_forecast <- forecast(d1982ETS\_MNN, h = 14)

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast(d1982ETS\_MNN\_forecast, h=14))

# CHECK TREND

# Fit Holt’s Method (ETS(A,A,N)) to the training data

# using the 'AAN' (Additive error, Additive trend, no seasonality) model

d1982ets\_AAN <- ets(d1982train, model="AAN", damped=TRUE)

summary(d1982ets\_AAN)

# using the 'MMN' (Multiplicative error, Multiplicative trend, no seasonality) model

d1982ets\_MMN <- ets(d1982train, model="MMN", damped=TRUE)

summary(d1982ets\_MMN)

# Generate a forecast for the next 14 time periods using Holt's Method

forecast.ets(d1982ets\_MMN, h=14)$mean

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast.ets(d1982ets\_MMN, h=14))

# CHECK SEASONALITY

# Fit Holt-Winter (ETS(A,A,A)) to the training data

# using the 'AAA' (Additive error, Additive trend, Additive seasonality) model

d1982ets\_AAA <- ets(d1982train, model="AAA", damped=TRUE)

summary(d1982ets\_AAA)

# Fit the ETS model with multiplicative seasonality (ETS(M,A,M))

# using the 'MAM' (Multiplicative error, Additive trend, Multiplicative seasonality) model

d1982ets\_MAM <- ets(d1982train, model="MAM", damped=TRUE)

summary(d1982ets\_MAM)

# Generate a forecast for the next 14 time periods using Holt's Method

forecast.ets(d1982ets\_AAA, h=14)$mean

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast.ets(d1982ets\_AAA, h=14))

# Manually select

d1982ets\_MAA <- ets(d1982train, model="MAA", damped=TRUE)

summary(d1982ets\_MAA)

# Generate a forecast for the next 14 time periods using Holt's Method

forecast.ets(d1982ets\_MAA, h=14)$mean

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast.ets(d1982ets\_MAA, h=14))

# Selecting the best model Automatically

d1982ets\_ZZZ <- ets(d1982\_ts, model="ZZZ")

summary(d1982ets\_ZZZ)

plot(forecast.ets(d1982ets\_ZZZ))

#1983

# Read data files 1983

d1983 <- read\_csv("1983.csv") # Read the data for the year 1983

# Convert to time series objects

d1983\_ts <- ts(d1983, frequency = 12, start = c(1983,1)) # Convert 1983 data to a time series object with monthly frequency

# Forecast Model Building

# Find the total number of observations

d1983length <- length(d1983\_ts) # Get the number of observations in the 1983 time series

# Write down size of train set

d1983train\_length <- 120 # Define the size of the training set (first 130 observations)

# Define the forecasting horizon

d1983h <- 14 # Define the forecasting horizon (14 steps ahead)

# Create the training set

d1983train <- ts(d1983\_ts[1:d1983train\_length], frequency = 12, start = c(1983, 1))

# Create the test set

d1983test <- ts(d1983\_ts[(d1983train\_length + 1):d1983length], frequency = 12, start = c(1983, (d1983train\_length %/% 12) + 1))

# CHECK ERROR

# Fit an ETS model with additive error, trend, and seasonality components

# using the 'ANN' (Additive error, no trend, no seasonality) model

# using the 'MNN' (Multiplicative error, no trend, no seasonality) model

d1983ETS\_ANN <- ets(d1983train, model = "ANN")

d1983ETS\_MNN <- ets(d1983train, model = "MNN")

summary(d1983ETS\_ANN)

summary(d1983ETS\_MNN)

# Generate the forecast using the ETS model for 14 steps ahead

d1983ETS\_MNN\_forecast <- forecast(d1983ETS\_MNN, h = 14)

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast(d1983ETS\_MNN\_forecast, h=14))

# CHECK TREND

# Fit Holt’s Method (ETS(A,A,N)) to the training data

# using the 'AAN' (Additive error, Additive trend, no seasonality) model

d1983ets\_AAN <- ets(d1983train, model="AAN", damped=TRUE)

summary(d1983ets\_AAN)

# using the 'MMN' (Multiplicative error, Multiplicative trend, no seasonality) model

d1983ets\_MMN <- ets(d1983train, model="MMN", damped=TRUE)

summary(d1983ets\_MMN)

# Generate a forecast for the next 14 time periods using Holt's Method

forecast.ets(d1983ets\_MMN, h=14)$mean

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast.ets(d1983ets\_MMN, h=14))

# CHECK SEASONALITY

# Fit Holt-Winter (ETS(A,A,A)) to the training data

# using the 'AAA' (Additive error, Additive trend, Additive seasonality) model

d1983ets\_AAA <- ets(d1983train, model="AAA", damped=TRUE)

summary(d1983ets\_AAA)

# Fit the ETS model with multiplicative seasonality (ETS(M,A,M))

# using the 'MAM' (Multiplicative error, Additive trend, Multiplicative seasonality) model

d1983ets\_MAM <- ets(d1983train, model="MAM", damped=TRUE)

summary(d1983ets\_MAM)

# Generate a forecast for the next 14 time periods using Holt's Method

forecast.ets(d1983ets\_MAM, h=14)$mean

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast.ets(d1983ets\_MAM, h=14))

# Manually

# using the 'MAM' (Multiplicative error, Additive trend, Multiplicative seasonality) model

d1983ets\_MMM <- ets(d1983train, model="MMM", damped=TRUE)

summary(d1983ets\_MMM)

# Generate a forecast for the next 14 time periods using Holt's Method

forecast.ets(d1983ets\_MMM, h=14)$mean

# Plot the forecast generated by the ETS model for 14 steps ahead

plot(forecast.ets(d1983ets\_MMM, h=14))

# Selecting the best model Automatically

d1983ets\_ZZZ <- ets(d1983\_ts, model="ZZZ")

summary(d1983ets\_ZZZ)

plot(forecast.ets(d1983ets\_ZZZ))

plot(forecast.ets(d1982ets\_ZZZ))

1. **ARIMA Model**

# Load necessary libraries

library(tsutils) # For time series utility functions

library(forecast) # For forecasting functions

library(tseries) # For time series analysis functions

library(readr) # For reading CSV files

library(smooth) # For smoothing

# 1982

# Read data files 1982

d1982 <- read\_csv("1982.csv") # Read the data for the year 1982

# Convert to time series objects

d1982\_ts <- ts(d1982, frequency = 12, start = c(1982,1)) # Convert 1982 data to a time series object with monthly frequency

# Forecast Model Building

# Find the total number of observations

d1982length <- length(d1982\_ts) # Get the number of observations in the 1982 time series

# Write down size of train set

d1982train\_length <- 130 # Define the size of the training set (first 130 observations)

# Define the forecasting horizon

d1982h <- 14 # Define the forecasting horizon (14 steps ahead)

# Create the training set

d1982train <- ts(d1982\_ts[1:d1982train\_length], frequency = 12, start = c(1982, 1))

# Create the test set

d1982test <- ts(d1982\_ts[(d1982train\_length + 1):d1982length], frequency = 12, start = c(1982, (d1982train\_length %/% 12) + 1))

# Check for the need for seasonal differencing

d1982seasonal <- nsdiffs(d1982train)

# Perform training set differencing

diff\_d1982train <- diff(d1982train)

plot(diff\_d1982train, main = "Differenced Training Set 1982")

# Check stationarity after differencing

adf.test(diff\_d1982train)

# Perform seasonal differencing

diff\_seasonal\_d1982train <- diff(d1982train, lag = 12, differences = 1)

plot(diff\_seasonal\_d1982train, main = "Seasonally Differenced Training Set 1982")

# Check stationarity after differencing

adf.test(diff\_seasonal\_d1982train)

# Perform second seasonal differencing

diff\_seasonal\_d1982train2 <- diff(diff\_seasonal\_d1982train, lag = 12, differences = 1)

plot(diff\_seasonal\_d1982train2, main = "Twice Seasonally Differenced Training Set 1982")

# Check stationarity after second seasonal differencing

adf.test(diff\_seasonal\_d1982train2)

# Display ACF and PACF for differenced training set

tsdisplay(diff\_d1982train, main = "ACF and PACF of Differenced Training Set 1982")

# Display ACF and PACF for twice seasonally differenced training set

tsdisplay(diff\_seasonal\_d1982train2, main = "ACF and PACF of Twice Seasonally Differenced Training Set 1982")

# ARIMA model

ARIMA\_1982train <- Arima(d1982train, order = c(0, 1, 0), seasonal = c(1, 2, 2))

# Display residuals of the ARIMA model

tsdisplay(residuals(ARIMA\_1982train), main = "Residuals of ARIMA Model 1982")

# Forecast using the ARIMA model

fcARIMA\_1982train <- forecast(ARIMA\_1982train, h = 14)$mean

# Plot the forecast

plot(forecast(ARIMA\_1982train, h = 14), main = "ARIMA Forecast 1982")

lines(d1982test, col = "red", type = "o")

# Align the forecasted mean values with the test set time indices

fcARIMA\_1982train <- ts(fcARIMA\_1982train, start = start(d1982test), frequency = frequency(d1982test))

# Calculate the forecast errors

fcARIMA\_1982train\_error <- d1982test - fcARIMA\_1982train

# Calculate and print summary statistics of the forecast errors

fcARIMA\_1982train\_ME <- mean(fcARIMA\_1982train\_error)

fcARIMA\_1982train\_MSE <- mean(fcARIMA\_1982train\_error^2)

fcARIMA\_1982train\_MAE <- mean(abs(fcARIMA\_1982train\_error))

fcARIMA\_1982train\_MAPE <- 100 \* mean(abs(fcARIMA\_1982train\_error) / d1982test)

print(paste("Mean Error (ME): ", fcARIMA\_1982train\_ME))

print(paste("Mean Squared Error (MSE): ", fcARIMA\_1982train\_MSE))

print(paste("Mean Absolute Error (MAE): ", fcARIMA\_1982train\_MAE))

print(paste("Mean Absolute Percentage Error (MAPE): ", fcARIMA\_1982train\_MAPE))

# AUTO ARIMA

AUTOARIMA\_1982train <- auto.arima(d1982train)

tsdisplay(residuals(AUTOARIMA\_1982train))

# Forecast using the ARIMA model

AUTO\_fcARIMA\_1982train <- forecast(ARIMA\_1982train, h = 14)$mean

# Plot the forecast

plot(forecast(AUTOARIMA\_1982train, h = 14), main = "AUTO ARIMA Forecast 1982")

lines(d1982test, col = "red", type = "o")

# Align the forecasted mean values with the test set time indices

AUTO\_fcARIMA\_1982train <- ts(AUTO\_fcARIMA\_1982train, start = start(d1982test), frequency = frequency(d1982test))

# Calculate the forecast errors

AUTO\_fcARIMA\_1982train\_error <- d1982test - AUTO\_fcARIMA\_1982train

# Calculate and print summary statistics of the forecast errors

AUTO\_fcARIMA\_1982train\_ME <- mean(AUTO\_fcARIMA\_1982train\_error)

AUTO\_fcARIMA\_1982train\_MSE <- mean(AUTO\_fcARIMA\_1982train\_error^2)

AUTO\_fcARIMA\_1982train\_MAE <- mean(abs(AUTO\_fcARIMA\_1982train\_error))

AUTO\_fcARIMA\_1982train\_MAPE <- 100 \* mean(abs(AUTO\_fcARIMA\_1982train\_error) / d1982test)

print(paste("Mean Error (ME): ", AUTO\_fcARIMA\_1982train\_ME))

print(paste("Mean Squared Error (MSE): ", AUTO\_fcARIMA\_1982train\_MSE))

print(paste("Mean Absolute Error (MAE): ", AUTO\_fcARIMA\_1982train\_MAE))

print(paste("Mean Absolute Percentage Error (MAPE): ", AUTO\_fcARIMA\_1982train\_MAPE))

#1983

# Read data files 1983

d1983 <- read\_csv("1983.csv") # Read the data for the year 1983

# Convert to time series objects

d1983\_ts <- ts(d1983, frequency = 12, start = c(1983,1)) # Convert 1983 data to a time series object with monthly frequency

# Forecast Model Building

# Find the total number of observations

d1983length <- length(d1983\_ts) # Get the number of observations in the 1983 time series

# Write down size of train set

d1983train\_length <- 120 # Define the size of the training set (first 130 observations)

# Define the forecasting horizon

d1983h <- 14 # Define the forecasting horizon (14 steps ahead)

# Create the training set

d1983train <- ts(d1983\_ts[1:d1983train\_length], frequency = 12, start = c(1983, 1))

# Create the test set

d1983test <- ts(d1983\_ts[(d1983train\_length + 1):d1983length], frequency = 12, start = c(1983, (d1983train\_length %/% 12) + 1))

# Check for the need for seasonal differencing

d1983seasonal <- nsdiffs(d1983train)

# Perform training set differencing

diff\_d1983train <- diff(d1983train)

plot(diff\_d1983train, main = "Differenced Training Set 1983")

# Check stationarity after differencing

adf.test(diff\_d1983train)

# Perform seasonal differencing

diff\_seasonal\_d1983train <- diff(d1983train, lag = 12, differences = 1)

plot(diff\_seasonal\_d1983train, main = "Seasonally Differenced Training Set 1983")

# Check stationarity after differencing

adf.test(diff\_seasonal\_d1983train)

# Display ACF and PACF for differenced training set

tsdisplay(diff\_d1983train, main = "ACF and PACF of Differenced Training Set 1983")

# Display ACF and PACF for seasonally differenced training set

tsdisplay(diff\_seasonal\_d1983train, main = "ACF and PACF of Seasonally Differenced Training Set 1983")

# ARIMA model

ARIMA\_1983train <- Arima(d1983train, order = c(1, 1, 1), seasonal = c(0, 1, 0))

# Display residuals of the ARIMA model

tsdisplay(residuals(ARIMA\_1983train), main = "Residuals of ARIMA Model 1983")

# Forecast using the ARIMA model

fcARIMA\_1983train <- forecast(ARIMA\_1983train, h = 14)$mean

# Plot the forecast

plot(forecast(ARIMA\_1983train, h = 14), main = "ARIMA Forecast 1983")

lines(d1983test, col = "red", type = "o")

# Align the forecasted mean values with the test set time indices

fcARIMA\_1983train <- ts(fcARIMA\_1983train, start = start(d1983test), frequency = frequency(d1983test))

# Calculate the forecast errors

fcARIMA\_1983train\_error <- d1983test - fcARIMA\_1983train

# Calculate and print summary statistics of the forecast errors

fcARIMA\_1983train\_ME <- mean(fcARIMA\_1983train\_error)

fcARIMA\_1983train\_MSE <- mean(fcARIMA\_1983train\_error^2)

fcARIMA\_1983train\_MAE <- mean(abs(fcARIMA\_1983train\_error))

fcARIMA\_1983train\_MAPE <- 100 \* mean(abs(fcARIMA\_1983train\_error) / d1983test)

print(paste("Mean Error (ME): ", fcARIMA\_1983train\_ME))

print(paste("Mean Squared Error (MSE): ", fcARIMA\_1983train\_MSE))

print(paste("Mean Absolute Error (MAE): ", fcARIMA\_1983train\_MAE))

print(paste("Mean Absolute Percentage Error (MAPE): ", fcARIMA\_1983train\_MAPE))

# AUTO ARIMA

AUTOARIMA\_1983train <- auto.arima(d1983train)

tsdisplay(residuals(AUTOARIMA\_1983train))

# Forecast using the ARIMA model

AUTO\_fcARIMA\_1983train <- forecast(ARIMA\_1983train, h = 14)$mean

# Plot the forecast

plot(forecast(AUTOARIMA\_1983train, h = 14), main = "AUTO ARIMA Forecast 1983")

lines(d1983test, col = "red", type = "o")

# Align the forecasted mean values with the test set time indices

AUTO\_fcARIMA\_1983train <- ts(AUTO\_fcARIMA\_1983train, start = start(d1983test), frequency = frequency(d1983test))

# Calculate the forecast errors

AUTO\_fcARIMA\_1983train\_error <- d1983test - AUTO\_fcARIMA\_1983train

# Calculate and print summary statistics of the forecast errors

AUTO\_fcARIMA\_1983train\_ME <- mean(AUTO\_fcARIMA\_1983train\_error)

AUTO\_fcARIMA\_1983train\_MSE <- mean(AUTO\_fcARIMA\_1983train\_error^2)

AUTO\_fcARIMA\_1983train\_MAE <- mean(abs(AUTO\_fcARIMA\_1983train\_error))

AUTO\_fcARIMA\_1983train\_MAPE <- 100 \* mean(abs(AUTO\_fcARIMA\_1983train\_error) / d1983test)

print(paste("Mean Error (ME): ", AUTO\_fcARIMA\_1983train\_ME))

print(paste("Mean Squared Error (MSE): ", AUTO\_fcARIMA\_1983train\_MSE))

print(paste("Mean Absolute Error (MAE): ", AUTO\_fcARIMA\_1983train\_MAE))

print(paste("Mean Absolute Percentage Error (MAPE): ", AUTO\_fcARIMA\_1983train\_MAPE))

1. **Reggresion**

# Load necessary libraries

library(tsutils) # For time series utility functions

library(forecast) # For forecasting functions

library(tseries) # For time series analysis functions

library(readr) # For reading CSV files

library(smooth) # For smoothing

library(carData)

library(car)

# Read data files 1982

d1982 <- read\_csv("1982.csv") # Read the data for the year 1982

# Convert to time series objects

d1982\_ts <- ts(d1982, frequency = 12, start = c(1982,1)) # Convert 1982 data to a time series object with monthly frequency

# Forecast Model Building

# Find the total number of observations

d1982length <- length(d1982\_ts) # Get the number of observations in the 1982 time series

# Write down size of train set

d1982train\_length <- 130 # Define the size of the training set (first 130 observations)

# Define the forecasting horizon

d1982h <- 14 # Define the forecasting horizon (14 steps ahead)

# Create the training set

d1982train <- ts(d1982\_ts[1:d1982train\_length], frequency = 12, start = c(1982, 1))

# Create the test set

d1982test <- ts(d1982\_ts[(d1982train\_length + 1):d1982length], frequency = 12, start = c(1982, (d1982train\_length %/% 12) + 1))

#Create Dummy Variables

D1 <- ts(rep(c(1,0,0,0,0,0,0,0,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D2 <- ts(rep(c(0,1,0,0,0,0,0,0,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D3 <- ts(rep(c(0,0,1,0,0,0,0,0,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D4 <- ts(rep(c(0,0,0,1,0,0,0,0,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D5 <- ts(rep(c(0,0,0,0,1,0,0,0,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D6 <- ts(rep(c(0,0,0,0,0,1,0,0,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D7 <- ts(rep(c(0,0,0,0,0,0,1,0,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D8 <- ts(rep(c(0,0,0,0,0,0,0,1,0,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D9 <- ts(rep(c(0,0,0,0,0,0,0,0,1,0,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D10 <- ts(rep(c(0,0,0,0,0,0,0,0,0,1,0,0), 11)[1:130], frequency = 12, start = c(1982,1))

D11 <- ts(rep(c(0,0,0,0,0,0,0,0,0,0,1,0), 11)[1:130], frequency = 12, start = c(1982,1))

D12 <- ts(rep(c(0,0,0,0,0,0,0,0,0,0,0,1), 11)[1:130], frequency = 12, start = c(1982,1))

d1982\_DV <- cbind(d1982train,D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12)

lm1982 <- lm(d1982train ~ D1 + D2 + D3 + D4 + D5 + D6 + D7 + D8 + D9 + D10 + D11, data=d1982\_DV)

summary(lm1982)

AIC(lm1982)

tsdisplay(residuals(lm1982))

vif(lm1982)

# Create lagged variables

L1\_1982train <- lag(d1982train, -1)

L2\_1982train <- lag(d1982train, -2)

L3\_1982train <- lag(d1982train, -3)

L4\_1982train <- lag(d1982train, -4)

d1982\_DV <- cbind(d1982\_DV, L1\_1982train, L2\_1982train, L3\_1982train, L4\_1982train)

# Remove NA values resulting from lagging

lagged\_data1982 <- data.frame(

d1982train = d1982train[5:130],

L1\_1982train = L1\_1982train[4:129],

L2\_1982train = L2\_1982train[3:128],

L3\_1982train = L3\_1982train[2:127],

L4\_1982train = L4\_1982train[1:126],

D1 = D1[5:130],

D2 = D2[5:130],

D3 = D3[5:130],

D4 = D4[5:130],

D5 = D5[5:130],

D6 = D6[5:130],

D7 = D7[5:130],

D8 = D8[5:130],

D9 = D9[5:130],

D10 = D10[5:130],

D11 = D11[5:130],

D12 = D12[5:130]

)

# Build regression model including lagged variables and dummy variables

LD\_1982train <- lm(d1982train ~ L1\_1982train + L2\_1982train + L3\_1982train + L4\_1982train +

D1 + D2 + D3 + D4 + D5 + D6 + D7 + D8 + D9 + D10 + D11,

data = lagged\_data1982)

summary(LD\_1982train)

AIC(LD\_1982train)

# Predict next 14 values

future\_data1982 <- data.frame(

L1\_1982train = d1982\_ts[131:144], # Adjusted range to start from 131

L2\_1982train = d1982\_ts[130:143], # Adjusted range to start from 130

L3\_1982train = d1982\_ts[129:142], # Adjusted range to start from 129

L4\_1982train = d1982\_ts[128:141], # Adjusted range to start from 128

D1 = rep(c(0,0,1,0,0,0,0,0,0,0,0,0), 2)[1:14],

D2 = rep(c(0,0,0,1,0,0,0,0,0,0,0,0), 2)[1:14],

D3 = rep(c(0,0,0,0,1,0,0,0,0,0,0,0), 2)[1:14],

D4 = rep(c(0,0,0,0,0,1,0,0,0,0,0,0), 2)[1:14],

D5 = rep(c(0,0,0,0,0,0,1,0,0,0,0,0), 2)[1:14],

D6 = rep(c(0,0,0,0,0,0,0,1,0,0,0,0), 2)[1:14],

D7 = rep(c(0,0,0,0,0,0,0,0,1,0,0,0), 2)[1:14],

D8 = rep(c(0,0,0,0,0,0,0,0,0,1,0,0), 2)[1:14],

D9 = rep(c(0,0,0,0,0,0,0,0,0,0,1,0), 2)[1:14],

D10 = rep(c(0,0,0,0,0,0,0,0,0,0,0,1), 2)[1:14],

D11 = rep(c(1,0,0,0,0,0,0,0,0,0,0,0), 2)[1:14],

D12 = rep(c(0,1,0,0,0,0,0,0,0,0,0,0), 2)[1:14])

predictions1982 <- predict(LD\_1982train, newdata = future\_data1982)

# Create a time series object for the predicted values

predicted\_values\_ts <- ts(predictions1982, start = c(1992, 12), frequency = 12) # Assuming the predictions start from May 1982

# Plot the predictions and actual values

plot(d1982\_ts, xlim = c(1982, 1994), ylim = range(c(d1982\_ts, predicted\_values\_ts)),

main = "Actual vs. Predicted Values for 1982",

xlab = "Year", ylab = "Value")

lines(predicted\_values\_ts, col = "red")

legend("topleft", legend = c("Actual", "Predicted"), col = c("black", "red"), lty = 1)

d1982\_error <- d1982test - predictions1982

ME1982 <- mean(d1982\_error)

MSE1982 <- mean(d1982\_error ^ 2)

MAE1982 <- mean(abs(d1982\_error))

MAPE1982 <- 100 \* mean(abs(d1982\_error)/d1982test)

#1983

# Read data files 1983

d1983 <- read\_csv("1983.csv") # Read the data for the year 1983

# Convert to time series objects

d1983\_ts <- ts(d1983, frequency = 12, start = c(1983,1)) # Convert 1983 data to a time series object with monthly frequency

# Forecast Model Building

# Find the total number of observations

d1983length <- length(d1983\_ts) # Get the number of observations in the 1983 time series

# Write down size of train set

d1983train\_length <- 120 # Define the size of the training set (first 130 observations)

# Define the forecasting horizon

d1983h <- 14 # Define the forecasting horizon (14 steps ahead)

# Create the training set

d1983train <- ts(d1983\_ts[1:d1983train\_length], frequency = 12, start = c(1983, 1))

# Create the test set

d1983test <- ts(d1983\_ts[(d1983train\_length + 1):d1983length], frequency = 12, start = c(1983, (d1983train\_length %/% 12) + 1))

#Create Dummy Variables

DD1 <- ts(rep(c(1,0,0,0,0,0,0,0,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD2 <- ts(rep(c(0,1,0,0,0,0,0,0,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD3 <- ts(rep(c(0,0,1,0,0,0,0,0,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD4 <- ts(rep(c(0,0,0,1,0,0,0,0,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD5 <- ts(rep(c(0,0,0,0,1,0,0,0,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD6 <- ts(rep(c(0,0,0,0,0,1,0,0,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD7 <- ts(rep(c(0,0,0,0,0,0,1,0,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD8 <- ts(rep(c(0,0,0,0,0,0,0,1,0,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD9 <- ts(rep(c(0,0,0,0,0,0,0,0,1,0,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD10 <- ts(rep(c(0,0,0,0,0,0,0,0,0,1,0,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD11 <- ts(rep(c(0,0,0,0,0,0,0,0,0,0,1,0), 11)[1:120], frequency = 12, start = c(1983,1))

DD12 <- ts(rep(c(0,0,0,0,0,0,0,0,0,0,0,1), 11)[1:120], frequency = 12, start = c(1983,1))

d1983\_DV <- cbind(d1983train,DD1,DD2,DD3,DD4,DD5,DD6,DD7,DD8,DD9,DD10,DD11,DD12)

lm1983 <- lm(d1983train ~ DD1 + DD2 + DD3 + DD4 + DD5 + DD6 + DD7 + DD8 + DD9 + DD10 + DD11, data=d1983\_DV)

summary(lm1983)

AIC(lm1983)

tsdisplay(residuals(lm1983))

vif(lm1982)

# Create lagged variables

L1\_1983train <- lag(d1983train, -1)

L2\_1983train <- lag(d1983train, -2)

L3\_1983train <- lag(d1983train, -3)

L4\_1983train <- lag(d1983train, -4)

d1983\_DV <- cbind(d1983\_DV, L1\_1983train, L2\_1983train, L3\_1983train, L4\_1983train)

# Remove NA values resulting from lagging

lagged\_data1983 <- data.frame(

d1983train = d1983train[5:120],

L1\_1983train = L1\_1983train[4:119],

L2\_1983train = L2\_1983train[3:118],

L3\_1983train = L3\_1983train[2:117],

L4\_1983train = L4\_1983train[1:116],

DD1 = DD1[5:120],

DD2 = DD2[5:120],

DD3 = DD3[5:120],

DD4 = DD4[5:120],

DD5 = DD5[5:120],

DD6 = DD6[5:120],

DD7 = DD7[5:120],

DD8 = DD8[5:120],

DD9 = DD9[5:120],

DD10 = DD10[5:120],

DD11 = DD11[5:120],

DD12 = DD12[5:120]

)

# Build regression model including lagged variables and dummy variables

LD\_1983train <- lm(d1983train ~ L1\_1983train + L2\_1983train + L3\_1983train + L4\_1983train +

DD1 + DD2 + DD3 + DD4 + DD5 + DD6 + DD7 + DD8 + DD9 + DD10 + DD11,

data = lagged\_data1983)

summary(LD\_1983train)

AIC(LD\_1983train)

# Predict next 14 values

future\_data1983 <- data.frame(

L1\_1983train = d1983\_ts[121:134], # Adjusted range to start from 121

L2\_1983train = d1983\_ts[120:133], # Adjusted range to start from 120

L3\_1983train = d1983\_ts[119:132], # Adjusted range to start from 119

L4\_1983train = d1983\_ts[118:131], # Adjusted range to start from 118

DD1 = rep(c(0,0,1,0,0,0,0,0,0,0,0,0), 2)[1:14],

DD2 = rep(c(0,0,0,1,0,0,0,0,0,0,0,0), 2)[1:14],

DD3 = rep(c(0,0,0,0,1,0,0,0,0,0,0,0), 2)[1:14],

DD4 = rep(c(0,0,0,0,0,1,0,0,0,0,0,0), 2)[1:14],

DD5 = rep(c(0,0,0,0,0,0,1,0,0,0,0,0), 2)[1:14],

DD6 = rep(c(0,0,0,0,0,0,0,1,0,0,0,0), 2)[1:14],

DD7 = rep(c(0,0,0,0,0,0,0,0,1,0,0,0), 2)[1:14],

DD8 = rep(c(0,0,0,0,0,0,0,0,0,1,0,0), 2)[1:14],

DD9 = rep(c(0,0,0,0,0,0,0,0,0,0,1,0), 2)[1:14],

DD10 = rep(c(0,0,0,0,0,0,0,0,0,0,0,1), 2)[1:14],

DD11 = rep(c(1,0,0,0,0,0,0,0,0,0,0,0), 2)[1:14],

DD12 = rep(c(0,1,0,0,0,0,0,0,0,0,0,0), 2)[1:14]

)

predictions1983 <- predict(LD\_1983train, newdata = future\_data1983)

# Create a time series object for the predicted values

predicted\_values\_ts1983 <- ts(predictions1983, start = c(1993, 1), frequency = 12) # Assuming the predictions start from March 1994

# Plot the predictions and actual values

plot(d1983\_ts, xlim = c(1983, 1994), ylim = range(c(d1983\_ts, predicted\_values\_ts)),

main = "Actual vs. Predicted Values for 1983",

xlab = "Year", ylab = "Value")

lines(predicted\_values\_ts1983, col = "red")

legend("topleft", legend = c("Actual", "Predicted"), col = c("black", "red"), lty = 1)

d1983\_error <- d1983test - predictions1983

ME1983 <- mean(d1983\_error)

MSE1983 <- mean(d1983\_error ^ 2)

MAE1983 <- mean(abs(d1983\_error))

MAPE1983 <- 100 \* mean(abs(d1983\_error)/d1983test)