Time Series Forecasting

**Code-**

# Time Series Forecasting with ARIMA and ETS - AirPassengers Dataset

# Load required libraries

library(ggplot2)

library(forecast)

library(tseries)

library(Metrics)

library(zoo) # For proper date handling

# Step 1: Load and visualize data

data("AirPassengers")

ap <- AirPassengers

# Convert to data frame for plotting

df <- data.frame(

Date = as.Date(as.yearmon(time(ap))),

Passengers = as.numeric(ap)

)

# Plot the time series

ggplot(df, aes(x = Date, y = Passengers)) +

geom\_line(color = "blue") +

labs(title = "Monthly Air Passengers (1949-1960)",

x = "Year", y = "Number of Passengers") +

theme\_minimal()

# Step 2: Confirm time series object

print(class(ap)) # "ts"

print(start(ap)) # 1949

print(end(ap)) # 1960

print(frequency(ap)) # 12 (monthly)

# Step 3: Check stationarity and transform if needed

cat("\nADF Test on Original Data:\n")

print(adf.test(ap)) # Likely non-stationary

# Log transformation and differencing

log\_ap <- log(ap)

diff\_log\_ap <- diff(log\_ap)

cat("\nADF Test on Differenced Log Data:\n")

print(adf.test(diff\_log\_ap)) # Should be stationary

# Plot transformed series

autoplot(diff\_log\_ap) +

ggtitle("Differenced Log AirPassengers") +

xlab("Year") + ylab("Log Differenced Passengers")

# Step 4: Fit ARIMA and ETS models

fit\_arima <- auto.arima(log\_ap)

fit\_ets <- ets(log\_ap)

cat("\nARIMA Model Summary:\n")

print(summary(fit\_arima))

cat("\nETS Model Summary:\n")

print(summary(fit\_ets))

# Step 5: Forecast next 24 months

forecast\_arima <- forecast(fit\_arima, h = 24)

forecast\_ets <- forecast(fit\_ets, h = 24)

# Plot forecasts in log scale

autoplot(forecast\_arima) +

ggtitle("ARIMA Forecast (Log Scale)") +

xlab("Year") + ylab("Log(Passengers)")

autoplot(forecast\_ets) +

ggtitle("ETS Forecast (Log Scale)") +

xlab("Year") + ylab("Log(Passengers)")

# Convert forecasts back to original scale (apply exp() only to components)

exp\_forecast\_arima <- forecast\_arima

exp\_forecast\_arima$mean <- exp(forecast\_arima$mean)

exp\_forecast\_arima$lower <- exp(forecast\_arima$lower)

exp\_forecast\_arima$upper <- exp(forecast\_arima$upper)

exp\_forecast\_ets <- forecast\_ets

exp\_forecast\_ets$mean <- exp(forecast\_ets$mean)

exp\_forecast\_ets$lower <- exp(forecast\_ets$lower)

exp\_forecast\_ets$upper <- exp(forecast\_ets$upper)

# Plot forecasts on original scale

autoplot(exp\_forecast\_arima) +

ggtitle("ARIMA Forecast (Original Scale)") +

xlab("Year") + ylab("Passengers")

autoplot(exp\_forecast\_ets) +

ggtitle("ETS Forecast (Original Scale)") +

xlab("Year") + ylab("Passengers")

# Step 6: Model evaluation

train <- window(log\_ap, end = c(1958, 12))

test <- window(log\_ap, start = c(1959, 1))

# Fit models on training data

fit\_arima\_train <- auto.arima(train)

fit\_ets\_train <- ets(train)

# Forecast for test period

fc\_arima <- forecast(fit\_arima\_train, h = length(test))

fc\_ets <- forecast(fit\_ets\_train, h = length(test))

# Evaluate using MAE and RMSE

mae\_arima <- mae(test, fc\_arima$mean)

rmse\_arima <- rmse(test, fc\_arima$mean)

mae\_ets <- mae(test, fc\_ets$mean)

rmse\_ets <- rmse(test, fc\_ets$mean)

# Print performance

cat("\nModel Performance on Test Data:\n")

cat("ARIMA - MAE:", round(mae\_arima, 4), "RMSE:", round(rmse\_arima, 4), "\n")

cat("ETS - MAE:", round(mae\_ets, 4), "RMSE:", round(rmse\_ets, 4), "\n")

**GRAPHS**

A graph showing the growth of passengers

AI-generated content may be incorrect.

A graph showing the growth of a stock market

AI-generated content may be incorrect.

A graph of a number of years

AI-generated content may be incorrect.

A graph showing the growth of the stock market

AI-generated content may be incorrect.

A graph of the year

AI-generated content may be incorrect.

A graph of a graph showing the time of a flight

AI-generated content may be incorrect.