```
##SARIMA#######
auto.arima(flu_ts_train,trace=T) #R language relatively optimal model recognition command # automatic order

ARIMA<-arima(flu_ts_train, c(1,0,0), seasonal=list(order=c(1,0,0),period=12))
summary (ARIMA)
##A Box-Ljung white-noise test
Box.test (ARIMA$residuals,type ="Ljung-Box")

# Get the fit values on the training set
fitted_values <- fitted(ARIMA)
print(fitted_values)
write.xlsx(fitted_values, file ="C:/Users/User/Desktop/ARIMA--X/2013-2021/ARIMA/The training set was fitted to the value.xlsx")
# count MSE
mse <- mean((flu_ts_train - fitted_values)^2)
# countMAE
mae <- mean(abs(flu_ts_train - fitted_values))
# countRMSE
rmse <- sqrt(mean((flu_ts_train - fitted_values)^2))


library(boot)
# Create a function to calculate the MSE
calc_MSE <- function(data, indices) {
  observed <- data[indices, "observed"]
  predicted <- data[indices, "predicted"]
  mean((observed - predicted)^2)
}

# Create a function to calculate the MAE
calc_MAE <- function(data, indices) {
  observed <- data[indices, "observed"]
  predicted <- data[indices, "predicted"]
  mean(abs(observed - predicted))
}

# Create a function to calculate the RMSE
calc_RMSE <- function(data, indices) {
  observed <- data[indices, "observed"]
  predicted <- data[indices, "predicted"]
  sqrt(mean((observed - predicted)^2))
}
```

```
data <- data.frame(observed = flu_ts_train, predicted = fitted_values)
# By specifying type = "all", we can calculate the confidence intervals for the MSE, MAE, and RMSE simultaneously
boot_MSE <- boot(data, calc_MSE, R = 1000)
boot_MAE <- boot(data, calc_MAE, R = 1000)
boot_RMSE <- boot(data, calc_RMSE, R = 1000)

# Output results
boot_MSE_ci <- boot.ci(boot_MSE, type = "all", conf = 0.95)
boot_MAE_ci <- boot.ci(boot_MAE, type = "all", conf = 0.95)
boot_RMSE_ci <- boot.ci(boot_RMSE, type = "all", conf = 0.95)

print("95% Confidence Interval for MSE:")
print(boot_MSE_ci)
print("95% Confidence Interval for MAE:")
print(boot_MAE_ci)
print("95% Confidence Interval for RMSE:")
print(boot_RMSE_ci)

##Fit values on the fitted test set
fit <- forecast(ARIMA)
write.xlsx(fit, file ="C:/Users/User/Desktop/ARIMA--X/2013-2021/ARIMA/The test set fits the value.xlsx")

# The data for 2022.1-2022.12 using the fitted ARIMA model
forecast_values <- forecast(ARIMA, h = 12)    # Forecast data for the next 12 months
# Save the prediction results as time-series objects
forecast_ts <- as.ts(forecast_values$mean)
# count MSE
mse1 <- mean((flu_ts_test - forecast_ts)^2)
# countMAE
mae1 <- mean(abs(flu_ts_test - forecast_ts))
# countRMSE
rmse1 <- sqrt(mean((flu_ts_test - forecast_ts)^2))

# Create a function to calculate the MSE
calc_MSE <- function(data, indices) {
  observed <- data[indices, "observed"]
  predicted <- data[indices, "predicted"]
  mean((observed - predicted)^2)
}

# Create a function to calculate the MAE
calc_MAE <- function(data, indices) {
```

```
  observed <- data[indices, "observed"]
  predicted <- data[indices, "predicted"]
  mean(abs(observed - predicted))
}

# Create a function to calculate the RMSE
calc_RMSE <- function(data, indices) {
  observed <- data[indices, "observed"]
  predicted <- data[indices, "predicted"]
  sqrt(mean((observed - predicted)^2))
}
data <- data.frame(observed = flu_ts_test, predicted = forecast_ts)
# By specifying type = "all", we can calculate the confidence intervals for the MSE, MAE, and RMSE simultaneously
boot_MSE <- boot(data, calc_MSE, R = 1000)
boot_MAE <- boot(data, calc_MAE, R = 1000)
boot_RMSE <- boot(data, calc_RMSE, R = 1000)

# Output results
boot_MSE_ci <- boot.ci(boot_MSE, type = "all", conf = 0.95)
boot_MAE_ci <- boot.ci(boot_MAE, type = "all", conf = 0.95)
boot_RMSE_ci <- boot.ci(boot_RMSE, type = "all", conf = 0.95)

print("95% Confidence Interval for MSE:")
print(boot_MSE_ci)
print("95% Confidence Interval for MAE:")
print(boot_MAE_ci)
print("95% Confidence Interval for RMSE:")
print(boot_RMSE_ci)

# Limit the prediction results to within 2022.1-2022.12forecast_ts <- window(forecast_ts, start = c(2022, 1))
# Draw the training set, prediction set and prediction results
plot(flu_ts, xlim = c(2013, 2023), ylim = c(min(flu_ts), max(flu_ts)), main = "Forecasts from ARIMA(1,0,0)(1,0,0)[12] ",
       ylab="Influenza cases",xlab="")
axis(1, at = seq(2013, 2022, 1), labels = seq(2013, 2022, 1))
lines(flu_ts,pch=1, lty=1,col = "grey", lwd = 2)    #    training set
lines(fitted_values, lty=1,col = "blue", lwd = 2)    # The training set was fitted to the value
lines(forecast_ts, lty=6,col = "blue", lwd = 2)    # Forecast results
legend("topleft", legend = c("Actual", "ARIMA fitted", "ARIMA forecast"), col = c("grey", "blue", "blue"),lty=c(1, 1, 6), lwd = 3)
```