R codes FTSA

Table of Contents

<u>Day 1</u>

#Loading relevant library that contains data set

library(fpp)

?wmurders

wmurders

#plotting the data set

plot(wmurders)

#by exploring the wmurders plot there is clear trend in the data between 1975 to 2004

#Therefore we have to separate that portion to create a time series

#creating a subset of the original time series data.

#It selects a window of data from years 25 to 55 of the original time series

After1975 = window(ts(wmurders),25,55)

#This code line converts the previously selected window of data (After1975) into a new time series object.

#It sets the start and end years for this time series.

#So, you are now specifically analyzing data from 1975 to 2004

After1975 = ts(After1975,start = 1975, end = 2004)

plot(After1975)

#Data

<u>Moving Average Approach</u>

#his code calculates a moving average with a window size of 5 for the After1975 time series.

#It computes the average value of the time series within a rolling window of 5 data points

MA_5 = ma(After1975,5)

plot(MA_5)

#Plot actual vs fitted values in MA_5 using matplot

#sequence starting from no 1 that's what the first 1 says

#length of the the sequence is the length of After1975

#and final 1 says the increment of the sequence is 1 by 1

t = seq(1,length(After1975),1)

?matplot

matplot(t,cbind(After1975,MA_5),type="l",col=c("red","green"),lty = c(1,1))

Regression Approach

t = seq(1,length(After1975),1)

#This line fits a linear regression model.

#It models After1975 as a function of t,

#effectively trying to find a linear relationship between the two variables.

#The result is stored in the variable fit.

fit = lm(After1975~t)

#After fitting the linear regression model,

#this line extracts the fitted or predicted values from the model and assigns them to the variable fits.

#These predicted values represent the values of After1975 that the linear regression model estimates based on the values of t.

fits <- fit$fitted.values

matplot(t,cbind(After1975,fits),type="l",col=c("red","green"),lty = c(1,1))

summary(fit)

Overall, this output provides a summary of a linear regression model that predicts "After1975" based on the variable "t." The model appears to be statistically significant, and "t" is a significant predictor of "After1975," with a negative relationship (as "t" increases, "After1975" tends to decrease).

```
Call:
lm(formula = After1975 ~ t)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.50430 -0.25364  0.00389  0.21426  0.73185 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  4.544490   0.125600  36.182  < 2e-16 ***
t           -0.045932   0.007075  -6.492 4.92e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3354 on 28 degrees of freedom
Multiple R-squared:  0.6009,	Adjusted R-squared:  0.5866 
F-statistic: 42.15 on 1 and 28 DF,  p-value: 4.918e-07
```

Day 2

Single Exponential Approaching Method

library(fpp)

plot(oil)

Apply Single exponential smoothing for oil data set

ses() is function to apply single exponential smoothing

h=12 is the no of periods that want to forecast in future

smoothing parameter alpha is 0.8

fitSES <- ses(oil, h=12 ,alpha=0.8)

creating a sequence for time

t = seq(1,length(oil),1)

#calling fitted values in exponential smoothing model

fitSES$fitted

Assigning fitted values into a variable called fitSESFitted

FitSESFitted <- fitSES$fitted

#plotting actual vs fitted values of exponential smoothing method using matplot

matplot(t,cbind(oil,FitSESFitted),type="l",col=c("red","green"),lty=c(1,1))

Double exponential smoothing approach

we apply double exponential approach for series that have trend but no seasonal trend

plot(ausair)

?ausair

fitHolt <- holt(ausair,h=12)

fitHolt$model

Results explanation

#holt(y=ausair,h=12)

#y represent the data set and h=12 represent that no future periods that are going to estimate

#smoothing constant alpha is 0.999 near to 1

#this alpha represent the weight that given to recent observations.

#In this case alpha near to 1 means model gave more weight for the recent observations

#Initial state

l=6.4015 represent the initial level

b=1.0834 represent the initial trend

sigma=1.6243 represent the standard deviation of the residuals

lower AIC and BIC values are better when it comes to model selection.

Triple Exponential approach

#data set shown both seasonal and trend variations

plot(austourists)

austourists

#data taken from quarterly from 1999 to 2010

#variance are changing over the time in the data set

fitHw = hw(austourists,seasonal = "multiplicative")

fitHwFitted <- fitHw$fitted

#creating a time sequence

t = seq(1,length(austourists),1)

```
#plot actual vs fitted values using matplot
matplot(t,cbind(austourists,fitHwFitted),type="l",col=c("red","green"),lty = c(1,1))
```

Day 3

library(fpp)

library(MLmetrics)

?beer

#monthly beer production from jan 1991 to aug 1995

beer

#plotting data set

plot(beer)

length(beer)

#there are 56 of data records in this data set

#dividing into training and testing data set

trainingSet = window(ts(beer),1,44)

testingSet =window(ts(beer),45,56)

trainingSet = ts(trainingSet,frequency = 12,start = c(1991,1))

testingSet =ts(testingSet,frequency = 12,start = c(1994,9))

plot(testingSet)

#to retrieve the models we use triple exponential smoothing

#In first case we consider the model as additive model

fit1 <- hw(trainingSet,seasonal = "additive")

#In second case we consider the model as multiplicative model and give alpha beta and gamma values manually

fit2 <- hw(trainingSet,seasonal = "multiplicative", alpha= 0.1, beta = 0.1, gamma = 0.5)

#In third case we consider the model as multiplicative and set automatic values given by R

fit3 <- hw(trainingSet,seasonal = "multiplicative")

#check the inaccuracies of the models

accuracy(fit1)

ME RMSE MAE MPE MAPE MASE ACF1

#Training set -0.4186668 7.954451 6.223062 -0.5433373 4.264735 0.6444595 -0.2744465

accuracy(fit2)

ME RMSE MAE MPE MAPE MASE ACF1

#Training set -0.3179051 9.536847 7.41566 -0.3572227 5.083843 0.7679648 -0.248388

accuracy(fit3)

ME RMSE MAE MPE MAPE MASE ACF1

#Training set -0.5567706 7.258765 6.006083 -0.5922874 4.102665 0.6219892 -0.285722

#by considering the MAPE values of the all model model 3 shows the minimal MAPE value. Therefore model 3 is the best model

#Now extracting the mean values of the fit3 model

fit3$mean

#check the fitted values of model 3

fit3$fitted

#check the residuals of the model 3

```
fit3$residuals

#creating the time sequence for the model 3 for plotting purposes
t <- seq(1,length(trainingSet),1)
#check the training set vs fitted values in model 3
matplot(t,cbind(trainingSet,fit3$fitted),type="l",col=c("red","green"),lty=c(1,1))

### Classical decomposition approach
classicalModel <- decompose(trainingSet,type="additive")
classicalModel

fittedVal <- classicalModel$seasonal+classicalModel$trend

t <- seq(1:length(trainingSet))
matplot(t,cbind(trainingSet,fittedVal),type="l",col=c("red","green"),lty=c(1,1))

fit <- stl(trainingSet,s.window = "periodic", robust = TRUE)
plot(fit)
fore <- forecast(fit)
plot(fore)

fittedVal = fit$time.series[,1]+fit$time.series[,2]
MAPEvalMA =MAPE(trainingSet,fittedVal)*100
MAPEvalMA
#This mean on the average model's forecasting values are differnce from actual values by 3.9%
#It implays that model is good for forecasting
```

Day 5

library(fpp)

library(forecast)

plot(austourists)

a <- acf(austourists,50)

#At lag 0 ACF always 1 because time series always co-related with it self at lag 0

#This ACF does not approach to suddenly when its lags are increasing therefore series doesn't appear as stationary

pacf(austourists,50)

#PACF also doesn't indicate a sharp drop after few lags. That's shows the series is not stationary

D1Y = diff(austourists,1)

D1Y

#In here we apply differencing.

#In this case the data set differencing with the previous consecutive value to make the series stationary

#This 1 in the diff funtion represent that we applied first order differencing

D4Y = diff(austourists,4)

D4Y

#In here we applied 4th order differencing higher order difference helps to remove seasonal and trend effects

DD1Y =diff(D1Y,1)

#Here we apply first order differencing into the D1Y, which is already defferenced by order 1

DD1Y =diff(diff(austourists,1),1)

#Here we apply differencing twice to original data set

#This is equal to calculate the second order differencing

plot(austourists)

#here we can see trend and seasoanlity as well

plot(D1Y)

#here we can see seasonality but no trend is there

plot(D4Y)

#Higher order differencing model (oder 4) indicates no seasonality and trend

D4Y = diff(austourists,4)

#Here we apply order 4 differencing to the original data set

D1D4Y = diff(D4Y,1)

#In here we calculated the first order differencing on the 4th oder diffrenced austourist data set

plot(D1D4Y)

D4D1Y = diff(D4Y,1)

Acf(austourists,50)

#In every 4th lag there is a significant difference can be appear. This shows seasonality effect

Acf(D4Y,50)

#There are no seasoanl patters can be appeared

Acf(D1Y,50)

#Seasonal Pattern can be appear

Acf(D4D1Y,50)

#Deseasonalized and Detrend

Box.test(austourists,12)

#L-jung box test for white noise in time series data

#test will examine whether there is any significant autocorrelation in the residuals

#of the time series at a lag of 12 time periods (in this case, 12 months).

#df =12 because model examine the auto correlations with lag 12 time periods

p value is almost near to zero this mean we can reject null hypothesis with strong evidence

#That mean model has significant auto correlation with residuals

This suggests that the model used to analyze this data may not adequately capture all of its underlying patterns

<u>Day 7 punsisi</u>

```
library(fpp)

#Description about dataset
?wmurders

plot(wmurders)
#Multiplicative Data set

length(wmurders)

#partition the dataset into training set and testing set where the last 12 values are taken as the testing set
trainingSet = window(ts(wmurders), 1, 43)
testingSet = window(ts(wmurders), 44, 55)

#Converting Training and Testing data in to time series data sets
trainingSet = ts(trainingSet, frequency = 12, start=1950)
testingSet = ts(testingSet,frequency = 12,start = 2004)
length(testingSet)

trainingSet
testingSet

# Make variance constant by transforming
#used when dealing with time series data with varying variances over time
LWmurd = log(trainingSet)
```

```
plot(LWmurd)

Acf(LWmurd,100)

# Differencing helps remove trends and seasonality.
#Identify the order of non-seasonal differencing
D1LWmurd = diff(LWmurd,1)
plot(D1LWmurd)
Acf(D1LWmurd,100)
Pacf(D1LWmurd)

Box.test (D1LWmurd, lag=12, type="Ljung")

#identify the time series point of data trend is changing
#Since the pattern cannot be identified with the entire dataset,consider data from 22nd observation
trainingSet = window(ts(wmurders),22,50)
testingSet = window(ts(wmurders),51,55)

trainingSet = ts(trainingSet, frequency = 12, start=1952)

plot(trainingSet)
length(trainingSet)

Acf(trainingSet,100)
Pacf(trainingSet,100)
```

```
#Parameter estimation
fit1 = Arima(trainingSet,order = c(1,0,0), include.mean=TRUE)


summary(fit1)
coeftest(fit1)
#autoregressive coefficient and the intercept are highly significant in this ARIMA(1,0,0) model
```

Day 7 Nethma

```
library(fpp)
plot(wmurders)

length(wmurders)
#length is 55

?wmurders
wmurders
#data taken from 1950 to 2004
#in monthly basis

#partition data set into test and training set

trainingSet <- window(ts(wmurders),1,43)
testingSet <- window(ts(wmurders),44,55)

#Converting the training and testing set into time series variables
trainingSet <- ts(trainingSet, frequency = 12, start = 1950)
testingSet <- ts(testingSet, frequency = 12, start = 2004)

plot(trainingSet)
#Plot shows variance is vary over the time periods

# make the variance constant by transforming
# This helps to make the variance constant over when we dealing with series which vary the variance over the time
LWmurd = log(trainingSet)
```

```
plot(LWmurd)
Acf(LWmurd) #Shows that series is not stationary

ndiffs(LWmurd)

## Identifying the order of non-seasonal differencing
D1LWmurd = diff(LWmurd,1)
plot(D1LWmurd)
acf(D1LWmurd)
pacf(D1LWmurd)

Box.test(D1LWmurd,lag=12, type = "Ljung")
#P value is greater than 0.05 that mean we have don't have enough evidence to reject null hypothesis(Ho)
#That mean model doesn't have significant autocorrelation in the residuals of the D1LWmurd

# Modeling using the recent set of data
trainingSet = window(ts(wmurders),22,50)
testingSet = window(ts(wmurders),51,55)
trainingSet =ts(trainingSet,frequency = 12,start=1952)

plot(trainingSet)
Acf(trainingSet,50)
pacf(trainingSet,50)

#Parameter estimation
fit1 = Arima(trainingSet,order=c(1,0,0), include.mean = TRUE)
summary(fit1)
coeftest(fit1)
```

Day 8

```
library(fpp)

library(fUnitRoots)


plot(chicken)

chicken

length(chicken)


#partitioning the data set

trainingSet= window(ts(chicken),22,60)

testingSet = window(ts(chicken),61,70)


#Converting both training and testing in to time series variable

trainingSet = ts(trainingSet,frequency = 12, start = 1945)

testingSet = ts(trainingSet, frequency = 12, start = 1946)


# Identifying the non seasonal differencing

D1chick= diff(trainingSet,1)

plot(D1chick)


adfTest(D1chick,lags = 1,type = c("c"))

acf(D1chick)

pacf(D1chick)

#Interpretation:

#The null hypothesis of the ADF test is that the time series is non-stationary.

#The alternative hypothesis is that the time series is stationary.

#Since the p-value is less than 0.01, you can reject the null hypothesis.

#In other words, there is strong evidence to suggest that the D1chick time series is stationary
```

Day 9

```
library(fpp)
plot(euretail)
ndiffs(euretail)
fiteu=stl(euretail,s.window = 'periodic',robust=TRUE)
plot(fiteu)
?euretail
length(euretail)
euretail
Acf(euretail,100)
Deuretail = diff(euretail,1)
acf(Deuretail,100)
D4D1 = diff(Deuretail,4)

Acf(D4D1,100)# 2nd lag is much close to the margin, so it is not consider as a cut off.
pacf(D4D1,100)# At the 3rd lag it is cut off as it is very close to zero
#as pacf cut off by 3rd lag
#if we consider pacf cut off we set ARIMA p as 1. (1,1,0) AR
#if we consider acf cut off we set as (0,1,1) MA.

# if the seasonal is negative in acf , we set as MA model
#1st seasonal lag is negative as the 4th lag is negative.

# if the 1st seasonal lag is positive, we decide into seasonal AR term.
# 4th lag is significant and the 8th lag is insignificant
# if the 2nd seasonal lag is insignificant: no need to consider the other lags.
```

```
# we need to know the all reasons fr why p,d,q and P D Q values as follows.

?Arima
fit1 = Arima(euretail,order=c(0,1,1),seasonal=c(0,1,1))
fit1
summary(fit1)
#MAPE is more meaningful as it gives a percentage.
coeftest(fit1)#to observe the esti,mated parameters, and p values.

#interprete the outputs.
#the p value of the model is 0.08095 and it is more than 0.05, there for this is significant.

Box.test(residuals(fit1))
#H0: errors are independent
#H1: errors are not independent.

#to derive the forecast values,
forecast(fit1, 4)
fitted(fit1)
```