BAN 5753 Week 13 Time Series Basic R Code

```
#install.packages('haven')
#install.packages('ggplot2')
#install.packages('forecast')
#install.packages('fpp2')
#install.packages('TTR')
#install.packages('dplyr')
#install.packages('MLmetrics')

library(haven)
library(ggplot2)
library(forecast)
library(fpp2)
library(TTR)
library(dplyr)
library(MLmetrics)

#Import data
ecommerce <- read_sas("C:\\Users\\12695\\OneDrive - Oklahoma A and M System\\BAN 5753\\Week
13\\ecommerce.sas7bdat")

class(ecommerce) #confirm class
glimpse(ecommerce)

#Convert date format from SAS to R
ecommerce$date <- as.Date(ecommerce$date, origin = '1960-01-01')

#Verify datatype
str(ecommerce)

#Prepare Time Series Object
ecommerce_ts <-ts(ecommerce$Ecommerce,start = c(1999, 1), end = c(2013, 1), frequency = 4)
plot(ecommerce_ts)

##############################################
#===========Time Series Analysis============#
# Look at horizontal data:              #
#   -Simple Moving Average               #
#   -Exponential Smoothing               #
# Assess some trend-based data:          #
#   -Trend-Adjusted Exponential Smoothing   #
#
##############################################
#=======================
# Simple Moving Average
#=======================
ecommerce_ma = ma(ecommerce_ts, order = 4, centre = FALSE)
plot(ecommerce_ts)
lines(ecommerce_ts, col = 'red')

#=======================
```

```
# Exponential Smoothing
#=======================
ecommerce_es = HoltWinters(ecommerce_ts, beta = FALSE, gamma = FALSE, alpha = 0.3)

#Predicted values
ecommerce_es$fitted

ecommerce_es2 = HoltWinters(ecommerce_ts, beta = FALSE, gamma = FALSE, alpha = 0.7)

layout(1:2)
plot(ecommerce_es, main='alpha = 0.3')
plot(ecommerce_es2, main='alpha = 0.7')

#Obtain estimate of alpha; do not provide a value for alpha
ecommerce_es3 = HoltWinters(ecommerce_ts, beta = FALSE, gamma = FALSE)

plot(ecommerce_es3)

#Forecast the model beyond the known range of data
ecommerce_es3_fore = forecast(ecommerce_es3, h=8)

#Look at forecasted values
plot(ecommerce_es3_fore)

#Assess constant variance
layout(1:1)
plot(ecommerce_es3_fore$residuals)
lines(c(0, 14), c(0, 0), col = 'red')

plotForecastErrors = function(forecasterrors,forecasttitle) {
 #Function provided by Avril Coghlan
 forecasterrors = na.omit(forecasterrors)
 # make a histogram of the forecast errors:
 mybinsize = IQR(forecasterrors) / 4
 mysd = sd(forecasterrors)
 mymin = min(forecasterrors) - mysd * 5
 mymax = max(forecasterrors) + mysd * 3
 # generate normally distributed data with mean 0 and standard deviation mysd
 mynorm <- rnorm(10000, mean = 0, sd = mysd)
 mymin2 <- min(mynorm)
 mymax2 <- max(mynorm)
 if (mymin2 < mymin) { mymin <- mymin2 }
 if (mymax2 > mymax) { mymax <- mymax2 }
 # make a red histogram of the forecast errors, with the normally distributed data overlaid:
 mybins <- seq(mymin, mymax, mybinsize)
 hist(forecasterrors, col = "red", freq = FALSE, breaks = mybins, main=forecasttitle)
 # freq=FALSE ensures the area under the histogram = 1
 # generate normally distributed data with mean 0 and standard deviation mysd
 myhist <- hist(mynorm, plot = FALSE, breaks = mybins)
 # plot the normal curve as a blue line on top of the histogram of forecast errors:
 points(myhist$mids, myhist$density, type = "l", col = "blue", lwd = 2)
}

#Assess normality of residuals
```

```
plotForecastErrors(ecommerce_es3_fore$residuals,'Assessing Normal Distribution')

#Assess accuracy
accuracy(ecommerce_es3_fore)


#==============================
# Trend Exponential Smoothing
#==============================
ecommerce_es4 = HoltWinters(ecommerce_ts,
                  alpha = 0.2,
                  beta = 0.4,
                  gamma = FALSE,
                  l.start = 17.6,
                  b.start = 1.04)

ecommerce_es4$fitted

ecommerce_es5 = HoltWinters(ecommerce_ts,
                  alpha = 0.2,
                  beta = 0.8,
                  gamma = FALSE,
                  l.start = 17.6,
                  b.start = 1.04)

ecommerce_es6 = HoltWinters(ecommerce_ts,
                  alpha = 0.7,
                  beta = 0.4,
                  gamma = FALSE,
                  l.start = 17.6,
                  b.start = 1.04)

ecommerce_es7 = HoltWinters(ecommerce_ts,
                  alpha = 0.7,
                  beta = 0.8,
                  gamma = FALSE,
                  l.start = 17.6,
                  b.start = 1.04)

par(mfrow = c(2, 2))
plot(ecommerce_es4, main='a=0.2, b=0.4')
plot(ecommerce_es5, main='a=0.2, b=0.8')
plot(ecommerce_es6, main='a=0.7, b=0.4')
plot(ecommerce_es7, main = 'a=0.7, b=0.8')

par(mfrow = c(1, 1))

# Allow the model to determine alpha and beta
ecommerce_es8 = HoltWinters(ecommerce_ts,
                  gamma = FALSE,
                  l.start = 17.6,
                  b.start = 1.04)

ecommerce_es8
```

```
par(mfrow = c(2, 1))
plot(ecommerce_es8)

ecommerce_es8_fore = forecast(ecommerce_es8, h = 8)

plot(ecommerce_es8_fore)

# Assess constant variance
par(mfrow = c(1, 1))
plot(ecommerce_es8_fore$residuals)
lines(c(3, 14), c(0, 0), col = 'red')

# Assess normal distribution
plotForecastErrors(ecommerce_es8_fore$residuals, 'Assessing Normal Distribution')

#Assess accuracy
accuracy(ecommerce_es8_fore)

#====================================
# Holt-Winters Exponential Smoothing
#====================================
ecommerce_ts_dc = decompose(ecommerce_ts)

plot(ecommerce_ts_dc)

# Plot with and without seasonality
# Remove season to use Trend-Adjusted Exponential Smoothing
ecommerce_ts_trend = ecommerce_ts - ecommerce_ts_dc$seasonal

ecommerce_es = HoltWinters(ecommerce_ts_trend,
                  gamma = FALSE)

# Leave season in the model
ecommerce_es2 = HoltWinters(ecommerce_ts,
                   gamma = FALSE)

# Forecast the next 8 periods for both
par(mfrow = c(2, 1))
ecommerce_es_fore = forecast(ecommerce_es, h = 8)
ecommerce_es_fore2 = forecast(ecommerce_es2, h = 8)

plot(ecommerce_es_fore)
plot(ecommerce_es_fore2)

# Assess constant variance
par(mfrow = c(2, 1))
plot(ecommerce_es_fore$residuals, main='Ecommerce: No Seasonal Component')
lines(c(1946, 1960), c(0, 0), col = 'red')

plot(ecommerce_es_fore2$residuals, main='Ecommerce: With Seasonal Component')
lines(c(1946, 1960), c(0, 0), col = 'red')

#Assess normal distribution
```

```
plotForecastErrors(ecommerce_es_fore$residuals,'Ecommerce: No Seasonal Component')

plotForecastErrors(ecommerce_es_fore2$residuals, 'Ecommerce: With Seasonal Component')

#### Result: By removing season, the model fits better for
#### Trend-Adjusted Exponential Smoothing
#### Information is lost; conduct a Holt-Winters model instead.

#Assess accuracy
accuracy(ecommerce_es_fore)
accuracy(ecommerce_es_fore2)

#Ecommerce data in a Holt-Winters model
par(mfrow = c(1, 1))
ecommerce_es3 = HoltWinters(ecommerce_ts)

ecommerce_es3
plot(ecommerce_es3)

ecommerce_es3_fore = forecast(ecommerce_es3, h = 40)

plot(ecommerce_es3_fore, main='Forecast for 40 Periods')

#Autocorrelation assessment
Box.test(ecommerce_es3_fore$residuals, lag = 20, type = "Ljung-Box")

acf(na.omit(ecommerce_es3_fore$residuals))

#Assess accuracy
accuracy(ecommerce_es3_fore)
```