Data Analytics

**Time Series Analysis – Assignment - 3**

Name: - Ashish

Roll no.: - 17BCS006

=============================================================================

|  |
| --- |
| > # Loading the Required Libraries  > library(readr)  > library(ggfortify)  > library(tseries)  > library(Metrics)  > library(ggplot2)  > library(forecast)  > library(TTR)  > library(dplyr)  > require(graphics)  >  > yahoo=read.csv("C:/Users/Ashish/Downloads/yahoo.csv",header = T,  stringsAsFactors = F)  > class(yahoo)  [1] "data.frame"  > colnames(yahoo)  [1] "Date" "Open" "High" "Low" "Close" "Adj.Close" "Volume"  > head(yahoo)  Date Open High Low Close Adj.Close Volume  1 2019-05-03 17.97 18.72 17.84 18.30 18.05834 178500  2 2019-05-06 17.96 18.74 17.96 18.61 18.36425 263100  3 2019-05-07 18.40 18.75 18.35 18.69 18.44319 179400  4 2019-05-08 18.63 19.06 18.54 18.74 18.49253 82700  5 2019-05-09 18.58 19.26 18.50 19.22 18.96619 89700  6 2019-05-10 19.05 19.50 19.05 19.43 19.24853 72800  > class(yahoo$Date)  [1] "character"  > class(yahoo$Adj.Close)  [1] "numeric"  > which(yahoo$Date == "2019-05-03")  [1] 1  > which(yahoo$Date == "2020-05-01")  [1] 252  > yahoo[1:252,]    .  .  .  141 2019-11-20 24.06 24.33 24.00 24.20 24.12607 125200  142 2019-11-21 24.31 24.35 23.95 23.99 23.91671 36000  [ reached 'max' / getOption("max.print") -- omitted 110 rows ]  > # Q.2. Build a timeSeries object with the data.  > yahoo.ts = ts(data = yahoo$Adj.Close, frequency = 12,  start = c(2019-05-03), end=c(2020-05-01))  > class(yahoo.ts)  [1] "ts"  > str(yahoo.ts)  Time-Series [1:37] from 2011 to 2014: 18.1 18.4 18.4 18.5 19 ...  > yahoo.ts    > start(yahoo.ts)  [1] 2011 1  > end(yahoo.ts)  [1] 2014 1  > #The time() function extracts the time index as a ts object  > time(yahoo.ts)    > #The frequency per period and time interval between observations of a ts object  > frequency(yahoo.ts)  [1] 12  > deltat(yahoo.ts)  [1] 0.08333333  > #This will plot a time series of the data.  > par(mar=c(1,1,1,1))  > plot(yahoo.ts, col="blue", lwd=2, ylab="Adjusted close",  main="Monthly closing price of SBUX")  > #plot a subset of the data use the window() function inside of plot()  > par(mar=c(1,1,1,1))  > plot(window(yahoo.ts, start = c(2019-05-03), end=c(2020-05-01),  ylab="Adjusted close",col="blue", lwd=2, main="Monthly closing price of SBUX"))    > #This will fit a line (also called trend line) shown in below figure.  > par(mar=c(1,1,1,1))  > abline(reg = lm(yahoo.ts~time(yahoo.ts)),col="orange")    >  > #-----------------------------------------  > # Q.3. It will plot the yearly mean values  > par(mar=c(1,1,1,1))  > plot(aggregate(yahoo.ts,FUN = mean))    >  > #-----------------------------------------  > #Ques 4. Boxplot across Quarters of month  > boxplot(yahoo.ts~cycle(yahoo.ts),xlab="Date", ylab = "Adjusted close",  main ="Monthly closing price of SBUX")    > #Data Cleaning : Since in the above box plot there was an outlier so I use  tsclean() to remove the outlier.  > yahoo1=tsclean(yahoo.ts)  > boxplot(yahoo1~cycle(yahoo1), xlab="Date", ylab = "Adjusted close",  main ="Boxplot with no outliers")    #-----------------------------------------  > #Ques 5. Decomposition of Cleaned data (yahoo1) using stl function  > yahoo.ts\_stl <- stl(yahoo1, s.window = "periodic")  > #par(mar=c(1,1,1,1))  > plot(yahoo.ts\_stl, main = "Time series decompostion using stl")    >  > #------------------------------------------  > # Ques 6. Type of Seasonality : yearly since the period is 12.  > yahoo.ts\_stl\_seasonal <- yahoo.ts\_stl$time.series[,1]#seasonal  > plot((yahoo.ts\_stl\_seasonal),xlab="Data",ylab = "Adjusted close",  main="Seasonal plot")    > yahoo.ts\_stl\_trend <- yahoo.ts\_stl$time.series #tread  > plot((yahoo.ts\_stl\_trend),xlab="Data",ylab = "Adjusted close",main="Tread plot")  > yahoo.ts\_stl\_random <- yahoo.ts\_stl$time.series #random  > plot((yahoo.ts\_stl\_random),xlab="Data",ylab = "Adjusted close",main="Tread plot") |
|  |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  | | --- | | > #-------------------------------------------  > # Ques 7. Residue after removing seasonality and trend  > trend\_yahoo.ts = ma(yahoo.ts,order = 12, centre = T)  > plot(as.ts(yahoo.ts),col = "orange",xlab="Date", ylab = "Adjusted close",  main ="removing seasonality")    > lines(trend\_yahoo.ts)    > plot(as.ts(trend\_yahoo.ts),col = "black",xlab="Date", ylab = "Adjusted close",  main ="removing trend")    > # Removing trend from time series:  > Dtr = yahoo.ts/trend\_yahoo.ts  > plot(as.ts(Dtr),main="Time series after de-trending",col = "orange",xlab="Date",  ylab = "Adjusted close")    > #Removing seasonality  > m = t(matrix(data = trend\_yahoo.ts, nrow = 12))  > seayahoo.ts = colMeans(m,na.rm = T)  > plot((as.ts(rep(seayahoo.ts,4))), main ="Time series after de-seasonality")    > res = yahoo.ts/ (trend\_yahoo.ts \* seayahoo.ts)  > plot(as.ts(res),main="Residue after removing trending and seasonallity")    >  > #--------------------------------------------  > #Ques 8. Building Holt Winters Model and  > #Ques 12.Improving model by changing alpha,beta & gamma values  > #Trainig Dataset  > yahoo.ts\_train <- ts(yahoo1, frequency = 12, start = c(2019-05-03),  end=c(2020-05-01))  > yahoo.ts\_train    > #Model 1: Seasonal HoltWinters Model  > HWyahoo.ts <- HoltWinters(yahoo.ts\_train)  > HWyahoo.ts  Holt-Winters exponential smoothing with trend and additive seasonal component.  Call:  HoltWinters(x = yahoo.ts\_train)  Smoothing parameters:  alpha: 1  beta : 0  gamma: 0  Coefficients:  [,1]  a 18.211058941  b 0.004903408  s1 0.310162601  s2 0.376621267  s3 0.839753267  s4 -0.011802566  s5 -0.409715524  s6 -0.014329399  s7 -0.228724399  s8 -0.183961941  s9 -0.227967191  s10 -0.416528649  s11 -0.377517524  s12 0.344010059  > plot(HWyahoo.ts,main="Original time series agains the Predicted time series(ADD)")    > HWyahoo.ts <- HoltWinters(yahoo.ts\_train, seasonal ="multiplicative")  > HWyahoo.ts  Holt-Winters exponential smoothing with trend and multiplicative seasonal component.  Call:  HoltWinters(x = yahoo.ts\_train, seasonal = "multiplicative")  Smoothing parameters:  alpha: 1  beta : 0  gamma: 0  Coefficients:  [,1]  a 18.239660883  b 0.004903408  s1 1.015668389  s2 1.019186321  s3 1.043008989  s4 0.999454387  s5 0.978987776  s6 0.999436022  s7 0.988313476  s8 0.990611005  s9 0.988368376  s10 0.978848762  s11 0.980824064  s12 1.017292433  > plot(HWyahoo.ts, main = "Orignal time series against the Predictedmn Time  sries (MUL)")    > plot(fitted(HWyahoo.ts))    > #Model 2 : Simple Exponential Smoothing Model  > #This model has no trend and zero seasonal components  > Hotyahoo.ts <- HoltWinters(yahoo.ts\_train, beta = F, gamma = F)  > Hotyahoo.ts  Holt-Winters exponential smoothing without trend and without seasonal component.  Call:  HoltWinters(x = yahoo.ts\_train, beta = F, gamma = F)  Smoothing parameters:  alpha: 0.9999411  beta : FALSE  gamma: FALSE  Coefficients:  [,1]  a 18.55508  > Hotyahoo.ts$fitted #Store the forecasts made by HW    > plot(Hotyahoo.ts, main = "Original time series against the Predicted time series")    > #Model 3 : Non-seasonal Holt Winters  > #It is better Prediction model than hotyahoo.ts  > Hotyahoo.ts1 <- HoltWinters(yahoo.ts\_train, gamma =F)  > Hotyahoo.ts1  Holt-Winters exponential smoothing with trend and without seasonal component.  Call:  HoltWinters(x = yahoo.ts\_train, gamma = F)  Smoothing parameters:  alpha: 1  beta : 0.1607397  gamma: FALSE  Coefficients:  [,1]  a 18.55506900  b -0.06829953  > Hotyahoo.ts1$fitted    > plot(Hotyahoo.ts1,main="Original time series against the Predicted time series")    > #--------------------------------------  > #Ques 9. Predict the values for the next 25% of the time  > #Ques 10. Plot the predicted values along with the actual values to compare them.  > HW\_pred = predict(HWyahoo.ts, n.ahead = 3\*12)  > round(HW\_pred) # predicted values for next years  Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  2014 19 19 19 18 18 18 18 18 18 18 18  2015 19 19 19 19 18 18 18 18 18 18 18 18  2016 19 19 19 19 18 18 18 18 18 18 18 18  2017 19  > ts.plot(HW\_pred, yahoo1, col = "blue")    > H\_pred = predict(Hotyahoo.ts,n.ahead = 3\*12)  > ts.plot(yahoo1,H\_pred,col = "purple", main = "Original vs Predicted Values")    > # method 2 : Predict values using forecast()  > J\_Pred = forecast(Hotyahoo.ts,h=36)  >  > plot(J\_Pred)    > plot(J\_Pred$residuals)    > summary(Hotyahoo.ts$fitted)  xhat level  Min. :18.05 Min. :18.05  1st Qu.:18.62 1st Qu.:18.62  Median :19.05 Median :19.05  Mean :19.03 Mean :19.03  3rd Qu.:19.35 3rd Qu.:19.35  Max. :20.38 Max. :20.38  > # Model 3  > H\_pred1 = predict(Hotyahoo.ts1,n.ahead = 3\*12)  > data3 = round(H\_pred1) # predicted values for next 3 years  > data4 = round(tail(yahoo1,36),0) # original values  >  > ts.plot(yahoo1,H\_pred1,col = "navyblue", main = "Original vs Predicted Values") # dotted predicted lines    >  >  > # Plot of Predicted value vs Actual Quarterly Earnings for 3 years  > X = time(data3)  > Y1 = data.frame(data3)  > Y2 = data.frame(data4)  > df = tbl\_df(data.frame(X,Y1,Y2))  > df  # A tibble: 36 x 3  X fit data4  *<ts>* *<dbl>* *<ts>*  1 2014.083 18 18  2 2014.167 18 18  3 2014.250 18 18  4 2014.333 18 19  5 2014.417 18 19  6 2014.500 18 19  7 2014.583 18 19  8 2014.667 18 19  9 2014.750 18 19  10 2014.833 18 19  # ... with 26 more rows  >  > #par(mar=c(1,1,1,1))  > ggplot(df,aes(X))+  + geom\_line(aes(y=data3),colour='red')+  + geom\_line(aes(y=data4),colour='blue')+  + labs(y = "Earnings (in dollars)", x = "Data")+  + ggtitle("Comparison of predicted value of earnings and actual earnings")+  + scale\_y\_continuous(limits=c(0,40))    >  > # Visualization  > #par(mar=c(1,1,1,1))  > plot.ts(data3, col="red", type ="l",xlim=c(2011,2019),ylim=c(0,40),xlab = "DAta",  ylab = "Earning (in dollars)")  > par(new=TRUE)  > plot.ts(data4, col="blue",type="l",xlim=c(2011,2019),ylim=c(0,40),xlab = "DAta",  ylab = "Earning (in dollars)") | |  | | |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  | | --- | | > #-------------------------------------------  > # 11. RMSE (Root Mean Squared Error)  > # rmse(predicted,actual) Note: Here, data4 contains the actual values of 25% data  >  > # Model 1 :  > HWyahoo.ts$SSE  [1] 5.022137  > H\_rmse = rmse(HW\_pred,data4)  > H\_rmse  [1] 1.579646  >  > # Model 2 :  > Hotyahoo.ts$SSE  [1] 3.746099  > HW\_rmse = rmse(H\_pred,data4)  > HW\_rmse  [1] 4.792509  >  > # Model 3 :  > Hotyahoo.ts1$SSE  [1] 4.092141  > HW1\_rmse = rmse(H\_pred1,data4)  > HW1\_rmse  [1] 1.20867 | |  | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | #-------------------------------------  > # 13. Arima Model trained using 75% of the given data (i.e., yahoo.ts\_train)  > arimayahoo.ts <- auto.arima(yahoo.ts\_train)  > arimayahoo.ts  Series: yahoo.ts\_train  ARIMA(1,0,1) with non-zero mean  Coefficients:  ar1 ma1 mean  0.6334 0.6615 18.9280  s.e. 0.1500 0.1674 0.2098  sigma^2 estimated as 0.0928: log likelihood=-7.85  AIC=23.7 AICc=24.95 BIC=30.14  > # Since (p,d,q) = (0,1,1) We know that d = 0 => stationary data,  hence using auot.arima(), we get best arima model  > # Since d = 1, it will automatically differentiate the data  i.e., diff(yahoo1) once to get stationary data  > ggtsdiag(arimayahoo.ts)    > #-----------------------------------------  > # 14. Predicted Values for next 25% of time  > # Using predict()  > predicted = predict(arimayahoo.ts,n.ahead = 3\*12)  > predicted # predicted values for next 3 years    > ts.plot(yahoo1,predicted$pred,col = "green") # dotted predicted lines  >  > # Using forecast()  > Ar = forecast(arimayahoo.ts,h = 20)  > Ar # 2nd column shows the predicted value and rest 4 columns are predicted value under 80% and 95% confidence interval    > plot(Ar)    > JArPredict = forecast(arimayahoo.ts,h = 20)  > JArPredict  > #plot(JArPredict$residuals)  > #qqnorm(JArPredict$residuals)  >  > # Testing ARIMA model  > data1 = round(tail(predicted$pred,36),0) # predicted values  > data2 = round(tail(yahoo1,36),0) # original values  > data1  Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  2014 19 19 19 19 19 19 19 19 19 19 19  2015 19 19 19 19 19 19 19 19 19 19 19 19  2016 19 19 19 19 19 19 19 19 19 19 19 19  2017 19  > data2  Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  2011 18 18 18 19 19 19 19 19 19 19 19  2012 20 20 20 20 19 19 19 18 18 19 19 19  2013 19 19 18 19 19 19 19 19 19 19 19 19  2014 19   |  |  |  |  |  | | --- | --- | --- | --- | --- | | > #-----------------------------------------------------  > # 15.Visualization:Plot predicted values with actual values for better comparison  > x = time(data1)  > y1 = data.frame(data1)  > y2 = data.frame(data2)  > df = tbl\_df(data.frame(x,y1,y2))  > df  # A tibble: 36 x 3  x data1 data2  *<ts>* *<ts>* *<ts>*  1 2014.083 19 18  2 2014.167 19 18  3 2014.250 19 18  4 2014.333 19 19  5 2014.417 19 19  6 2014.500 19 19  7 2014.583 19 19  8 2014.667 19 19  9 2014.750 19 19  10 2014.833 19 19  # ... with 26 more rows  > # Plot of Predicted value vs Actual Quarterly Earnings for 3 years i.e., 2011-14  > ggplot(df,aes(x,hp))+  + geom\_line(aes(y=data1),colour='red')+  + geom\_line(aes(y=data2),colour='blue')+  + labs(y = "Earnings (in dollars)", x = "Time")+  + scale\_y\_continuous(limits=c(0,40))    > geom\_smooth(method="lm")  geom\_smooth: na.rm = FALSE, orientation = NA, se = TRUE  stat\_smooth: na.rm = FALSE, orientation = NA, se = TRUE, method = lm  position\_identity  > plot.ts(data1, col="red", type = "l",xlim=c(2011,2014),ylim=c(0,20),  ylab = "Earning (in dollars)")  > par(new=TRUE)  > plot(data2, col="blue",type="l",xlim=c(2011,12014),ylim=c(0,20),  ylab = "Earning(in dollars)"  ,main = "Predicted value vs Actual Quarterly Earnings for the year 1980")     |  | | --- | | > #------------------------------------------------  > # 16. Accuracy of ARIMA Model (using auto.arima())  > summary(arimayahoo.ts)  Series: yahoo.ts\_train  ARIMA(1,0,1) with non-zero mean  Coefficients:  ar1 ma1 mean  0.6334 0.6615 18.9280  s.e. 0.1500 0.1674 0.2098  sigma^2 estimated as 0.0928: log likelihood=-7.85  AIC=23.7 AICc=24.95 BIC=30.14  Training set error measures:  ME RMSE MAE MPE MAPE MASE ACF1  Training set 0.0169883 0.2920217 0.2231868 0.0666217 1.171267 0.2399256 -0.1523175  > accuracy(arimayahoo.ts) #: this is a function for arima model so to run this we  need to unload "Metrics" library  ME RMSE MAE MPE MAPE MASE ACF1  Training set 0.0169883 0.2920217 0.2231868 0.0666217 1.171267 0.2399256 -0.1523175  > rmse(data1,data2)  [1] NaN  > #------------------------------------------------  > # Ques17. Tuning the model by manually giving (p,d,q) values  > # p : AR (Auto-regressive model), d : I (Difference/ Integration part),  q : MA (Moving average time)  >  > # First make the Data Stationary  > plot(log(yahoo.ts)) # homongenizing variance    > abline(reg = lm(log(yahoo.ts)~time(yahoo.ts)),col="orange")    > m = diff(log(yahoo.ts))  > plot(m) # homogenizing mean    > # Preprocessing  > pacf(yahoo.ts) # q = 1    > pacf = acf(log(yahoo.ts))    > acf(yahoo.ts,plot =TRUE )    > # Each observation is positively associated with its recent past at  least through 4 lags.  > pacf(diff(log(yahoo.ts))) # p=3    > acf(m) # q =1    >  > # First Try :  > fit=arima(log(yahoo.ts\_train),c(3,1,1),seasonal = list(order=c(3,1,1),period=4))  > fit  Call:  arima(x = log(yahoo.ts\_train), order = c(2, 1, 1), seasonal = list(order = c(0,  1, 0), period = 12))  Coefficients:  ar1 ar2 ma1  -0.4076 -0.0917 0.7673  s.e. 0.2576 0.2190 0.1744  sigma^2 estimated as 0.0007023: log likelihood = 52.85, aic = -97.7  > pred = predict(fit,n.ahead = 3\*12)  > pred # next 3 ten years pred values(in log)    > pred1 = round(2.718^pred$pred,0)  > pred1  Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  2014 19 18 19 19 19 19 19 19 19 19 19  2015 18 19 18 18 19 19 19 19 19 19 19 19  2016 18 18 18 18 19 19 19 19 19 19 19 19  2017 18  > ts.plot(yahoo.ts,pred1,log="y",lty=c(1,3))#dotted predicted lines    >  > # Testing the model  > data11 = round(tail(pred1,20),0)#predicted values  > data22 = round(tail(yahoo.ts,20),0)#original values  > data11  Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  2015 19 19 19 19 19 19 19  2016 18 18 18 18 19 19 19 19 19 19 19 19  2017 18  > data22  Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  2012 19 19 18 18 19 19 19  2013 19 19 18 19 19 19 19 19 19 19 19 19  2014 19  > par(mfrow = c(2,1))  > # Visualization  > par(mar=c(1,1,1,1))  > plot.ts(data11, col="red", type = "l",ylim=c(0,20), ylab = "Earning (in dollars)", main = "Comparison of predicted value of earnings and actual earnings for 3 years 2011-14")  > par(new=TRUE)  > plot.ts(data22, col="blue",type="l",ylim=c(0,20), ylab = "Earning (in dollars)", main = "Comparison of predicted value of earnings and actual earnings for 3 years 2011-14")    >  > plot.ts(data1, col="red", type = "l",xlim=c(2014,2015),ylim=c(0,20),  ylab = "Earning (in dollars)", main = "Predicted values vs Actual Quarterly  Earnings for the year 2014")  > par(new=TRUE)  > plot(data2, col="blue",type="l",xlim=c(2014,2015),ylim=c(0,20),  ylab = "Earning (in dollars)", main = "Predicted values vs Actual Quarterly  Earnings for the year 2014") | |  | | |  | | --- | | > |   > # Accuracy of model  > # accuracy(fit)  > rmse(pred1,data2)  [1] 0.5  > accuracy(pred1,data2)  ME RMSE MAE MPE MAPE ACF1 Theil's U  Test set 0.25 0.5 0.25 1.315789 1.315789 0.287037 Inf  >  > # Second Try :  > yahoo.ts\_fit <- auto.arima(log(yahoo.ts\_train))  > plot(yahoo.ts\_fit)    > accuracy(yahoo.ts\_fit)  ME RMSE MAE MPE MAPE MASE  Training set 0.0009111758 0.01519777 0.01167771 0.02836161 0.396319 0.2398298  ACF1  Training set -0.1537559  > plot(accuracy(yahoo.ts\_fit)) |   > # 19. Raw data Vs Cleaned data  > par(mfrow = c(2,1))  > HW <- HoltWinters(yahoo.ts,seasonal = "multiplicative") # This will give same fitting as auto.arima()  > HW  Holt-Winters exponential smoothing with trend and multiplicative seasonal component.  Call:  HoltWinters(x = yahoo.ts, seasonal = "multiplicative")  Smoothing parameters:  alpha: 1  beta : 0  gamma: 0  Coefficients:  [,1]  a 18.239660883  b 0.004903408  s1 1.015668389  s2 1.019186321  s3 1.043008989  s4 0.999454387  s5 0.978987776  s6 0.999436022  s7 0.988313476  s8 0.990611005  s9 0.988368376  s10 0.978848762  s11 0.980824064  s12 1.017292433  > plot(HW, main = "Original time series against the Fitted time series : Raw Data")    > HWC <- HoltWinters(yahoo1,seasonal = "multiplicative") # This will give same fitting as auto.arima()  > HWC  Holt-Winters exponential smoothing with trend and multiplicative seasonal component.  Call:  HoltWinters(x = yahoo1, seasonal = "multiplicative")  Smoothing parameters:  alpha: 1  beta : 0  gamma: 0  Coefficients:  [,1]  a 18.239660883  b 0.004903408  s1 1.015668389  s2 1.019186321  s3 1.043008989  s4 0.999454387  s5 0.978987776  s6 0.999436022  s7 0.988313476  s8 0.990611005  s9 0.988368376  s10 0.978848762  s11 0.980824064  s12 1.017292433  > plot(HWC,main="Original time series against the Fitted time series:Cleaned Data")    > HW$SSE  [1] 5.022137  > HWC$SSE  [1] 5.022137 | |  | | |  | | --- | |  | | | | | | | |