Assignment 3 - Stock Forecasting

Quyen Hoang

2022-06-09

## Introduction

In this assignment, we will use 3 types of Exponential Smoothing Method (Simple. Double, Triple) and Linear Regression to forecast the stock price of three companies – Microsoft, Google, and Facebook in the past two and a half years to see which method yield the best result. Then, I use that model to forecast the stock price in short-term and long-term for three companies and choose the company to invest.

## Loading the Libraries

library(MLmetrics)

##   
## Attaching package: 'MLmetrics'

## The following object is masked from 'package:base':  
##   
## Recall

library(tidyquant)

## Loading required package: lubridate

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

## Loading required package: PerformanceAnalytics

## Loading required package: xts

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

##   
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':  
##   
## legend

## Loading required package: quantmod

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

library(fpp2)

## -- Attaching packages ---------------------------------------------- fpp2 2.4 --

## v ggplot2 3.3.5 v fma 2.4   
## v forecast 8.16 v expsmooth 2.3

##

options("getSymbols.warning4.0"=FALSE)  
options("getSymbols.yahoo.warning"=FALSE)  
  
  
### Downloading price using quantmod   
  
#Microsoft  
getSymbols("MSFT", from = '2020-01-01',  
 to = "2022-06-01",warnings = FALSE,  
 auto.assign = TRUE)

## [1] "MSFT"

options("getSymbols.warning4.0"=FALSE)  
options("getSymbols.yahoo.warning"=FALSE)  
  
#Google  
  
getSymbols("goog", from = '2020-01-01',  
 to = "2022-06-01",warnings = FALSE,  
 auto.assign = TRUE)

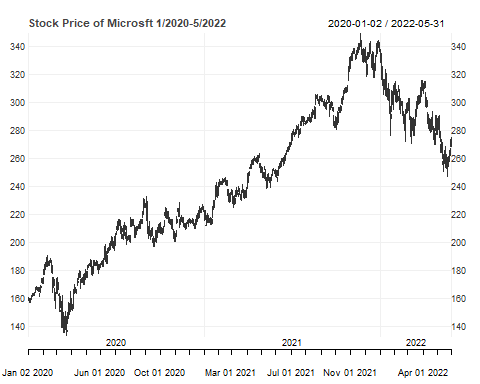
## [1] "goog"

#Facebook  
getSymbols("FB", from = '2020-01-01',  
 to = "2022-06-01",warnings = FALSE,  
 auto.assign = TRUE)

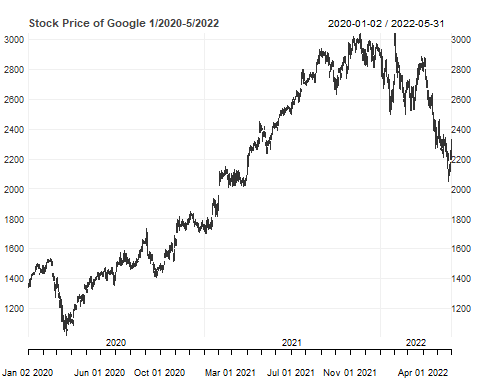
## [1] "FB"

## Chart Series

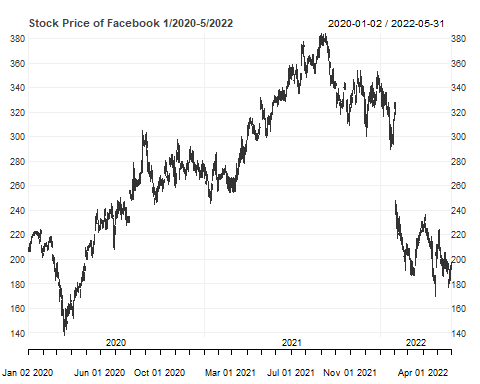
#MSFT  
chart\_Series(MSFT, name = "Stock Price of Microsft 1/2020-5/2022")



#Google  
chart\_Series(GOOG, name = "Stock Price of Google 1/2020-5/2022")



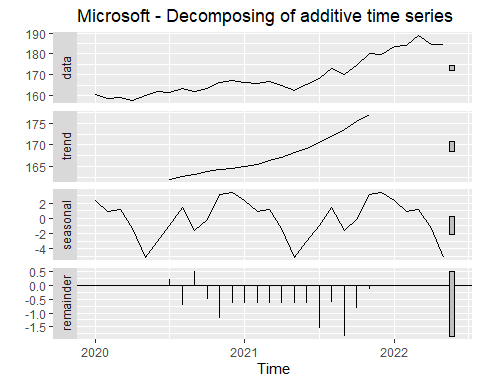
#Facebook  
chart\_Series(FB, name = "Stock Price of Facebook 1/2020-5/2022")



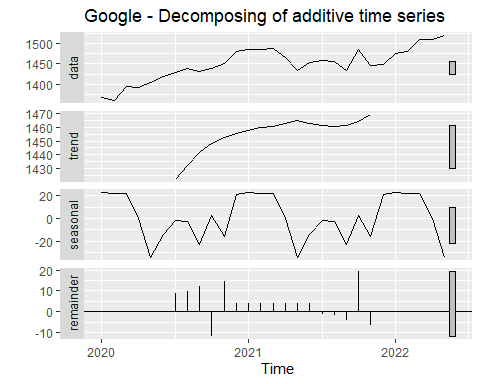
Overall, the stock price of three companies have upward trend until around the end of 2021 when they all started going down significantly. Especially the stock of FB whose price is as half of the peak time in 2021.

## Decompose

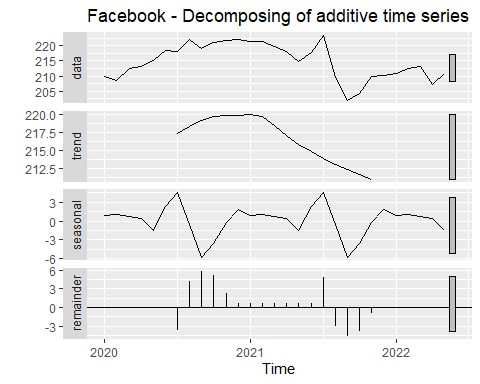
# Getting the Close Price Column  
tsmsft <- Cl(MSFT)  
tsgoog <- Cl(GOOG)  
tsfb<-Cl(FB)  
   
##Decompose  
#MSFT  
tsmsft1<-ts(tsmsft, start = c(2020,1), end =c(2022,5), frequency=12)  
dc.msft<-decompose(tsmsft1)  
autoplot(dc.msft)+ggtitle("Microsoft - Decomposing of additive time series")



#GOOG  
tsgoog1<-ts(tsgoog, start = c(2020,1), end =c(2022,5), frequency=12)  
dc.goog<-decompose(tsgoog1)  
autoplot(dc.goog)+ggtitle("Google - Decomposing of additive time series")



#FB  
tsfb1<-ts(tsfb, start = c(2020,1), end =c(2022,5), frequency=12)  
dc.fb<-decompose(tsfb1)  
autoplot(dc.fb)+ggtitle("Facebook - Decomposing of additive time series")



##Seasonal  
#MSFT  
dc.msft$seasonal

## Jan Feb Mar Apr May Jun  
## 2020 2.3105731 0.8355703 1.1222413 -1.3619335 -5.1698445 -3.1044192  
## 2021 2.3105731 0.8355703 1.1222413 -1.3619335 -5.1698445 -3.1044192  
## 2022 2.3105731 0.8355703 1.1222413 -1.3619335 -5.1698445   
## Jul Aug Sep Oct Nov Dec  
## 2020 -0.8904732 1.4436977 -1.5508842 -0.1285939 3.1639033 3.3301628  
## 2021 -0.8904732 1.4436977 -1.5508842 -0.1285939 3.1639033 3.3301628  
## 2022

#GOOG  
dc.goog$seasonal

## Jan Feb Mar Apr May Jun  
## 2020 22.6609979 22.3146745 22.1830824 0.1559376 -34.4085821 -14.2156288  
## 2021 22.6609979 22.3146745 22.1830824 0.1559376 -34.4085821 -14.2156288  
## 2022 22.6609979 22.3146745 22.1830824 0.1559376 -34.4085821   
## Jul Aug Sep Oct Nov Dec  
## 2020 -1.4715218 -2.8546649 -22.6896462 2.5353038 -15.4288634 21.2189109  
## 2021 -1.4715218 -2.8546649 -22.6896462 2.5353038 -15.4288634 21.2189109  
## 2022

#FB  
dc.fb$seasonal

## Jan Feb Mar Apr May Jun  
## 2020 0.8884059 1.0688278 0.7392325 0.3404889 -1.5253517 2.3933963  
## 2021 0.8884059 1.0688278 0.7392325 0.3404889 -1.5253517 2.3933963  
## 2022 0.8884059 1.0688278 0.7392325 0.3404889 -1.5253517   
## Jul Aug Sep Oct Nov Dec  
## 2020 4.4856916 -0.5378458 -5.8586796 -3.5609742 -0.2157608 1.7825691  
## 2021 4.4856916 -0.5378458 -5.8586796 -3.5609742 -0.2157608 1.7825691  
## 2022

### Microsoft

Trend: the long term movement of the data. In this plot, we can notice that MSFT stock has upward trend overall. I do not see any significant up trend of this stock. I would say there was just a slight tilt up started in the middle of 2021.

Seasonal: The closing price of MSFT tended to get the highest at the end of the year in December and the lowest in May. So overall, according to this pattern, we can say that the right time to sell the stock is at the end of the year (best time is December) and the right time to buy the stock is in the middle of the year (especially May).

### GOOG

Trend: Overall this company has uptrend stock price. Although there was a slight downward within the middle of 2021, it got back to the upward track afterward.

Seasonal: There is not much difference in the price of GOOG stock over the year. However, regarding to the pattern, the peak time is in January and the lowest is in May.

### FB

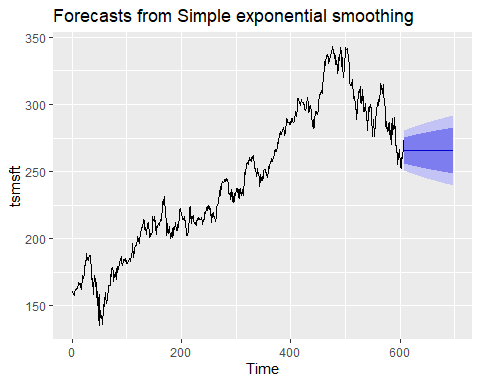
Trend: In this plot, we can see that Fb stock price’s trend was upward in 2020; however, it started going down in 2021 up until now. overall the downtrend is longer than the uptrend. Seasonal: Fb’s Stock Price seems to get the highest in July and the bottom in September. So the right time to buy is around September and October while the good time to sell is in July.

## Simple Exponential Smoothing

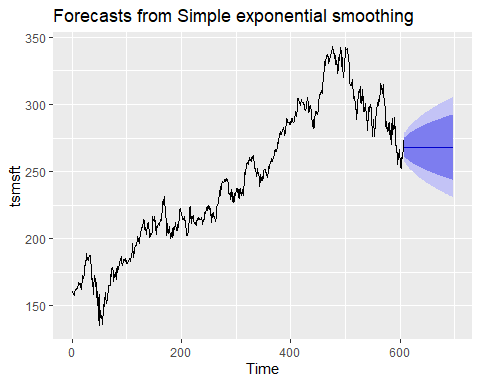
### MSFT

#PLot

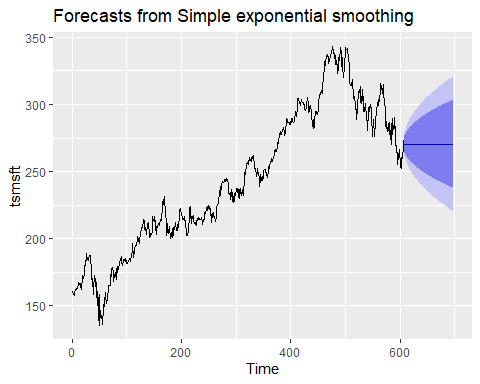
#alpha=0.15  
ses.msft1 <- ses(tsmsft, alpha = 0.15,h=90)  
autoplot(ses.msft1)



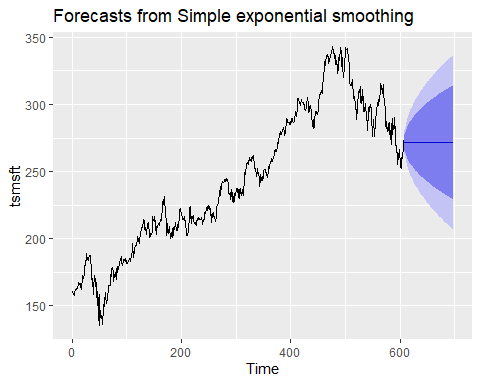
m1<-accuracy(ses.msft1)  
#alpha=0.35  
ses.msft2 <- ses(tsmsft, alpha = 0.35,h=90)  
autoplot(ses.msft2)



m2<-accuracy(ses.msft2)  
#alpha=0.55  
ses.msft3 <- ses(tsmsft, alpha = 0.55,h=90)  
autoplot(ses.msft3)



m3<-accuracy(ses.msft3)  
#alpha=0.75  
ses.msft4 <- ses(tsmsft, alpha = 0.75,h=90)  
autoplot(ses.msft4)



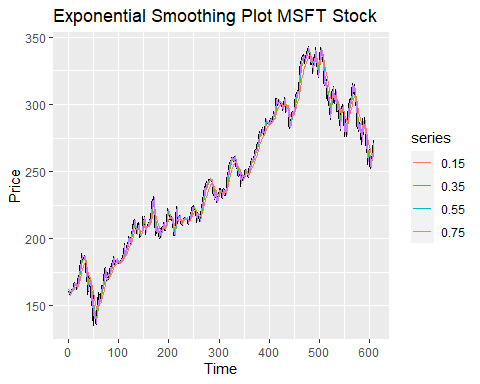
m4<-accuracy(ses.msft4)

#Compare Forecast with Actual time series.

##Accuracy Table  
m<-rbind(m1,m2,m3,m4)  
rownames(m)<-c("Alpha=0.15","Alpha=0.35","Alpha=0.55","Alpha=0.75")  
m

## ME RMSE MAE MPE MAPE MASE ACF1  
## Alpha=0.15 1.1352840 7.626221 6.071845 0.44000733 2.566233 1.7543421 0.78491616  
## Alpha=0.35 0.5082117 5.508393 4.259893 0.19464075 1.808335 1.2308138 0.53079490  
## Alpha=0.55 0.3311296 4.843867 3.686553 0.12371982 1.572622 1.0651581 0.28856567  
## Alpha=0.75 0.2446321 4.624645 3.460080 0.08842436 1.486788 0.9997233 0.06054824

##Plot  
autoplot(ts(tsmsft))+autolayer(ses.msft2$fitted, series ="0.35")+autolayer(ses.msft3$fitted, series ="0.55")+autolayer(ses.msft4$fitted, series ="0.75")+autolayer(ses.msft1$fitted, series = "0.15")+ggtitle("Exponential Smoothing Plot MSFT Stock")+labs(x="Time", y="Price")



Interpretation: - We can see the forecasting lines are flat which do not tell anything about the increasing trend as well as the seasonality of the time series

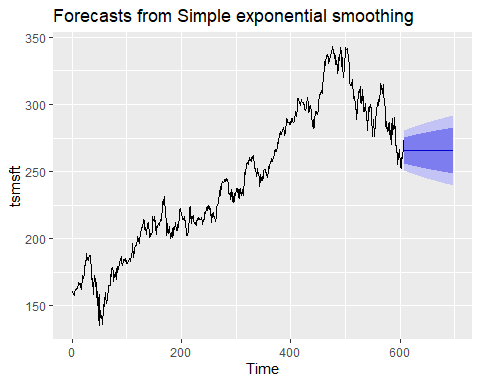
- According to the The accuracy tables, we can notice that the model with alpha=0.75 has the lowest error with all index (MSE, MAE, and MAPE). We conclude that alpha=0.75 gives out the best forecasting for all three companies.

- Again in the plot, we can see that the Purple line presenting for the forecast with alpha =0.75 is the closest to the black line which is the actual data. In contrast, the farthest is the red line which is of the forecasting with alpha=0.15

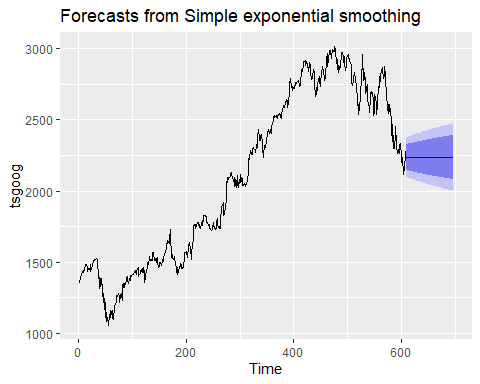
### 2. GOOGLE

#Plot

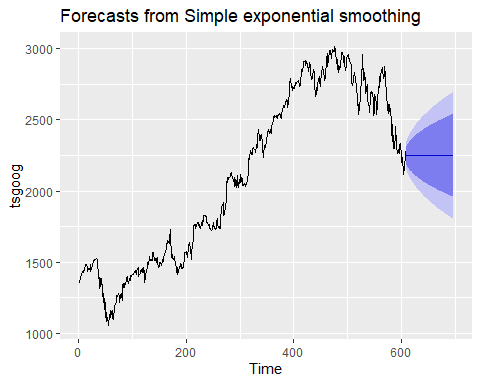
#alpha=0.15  
ses.goog1 <- ses(tsgoog, alpha = 0.15,h=90)  
autoplot(ses.msft1)



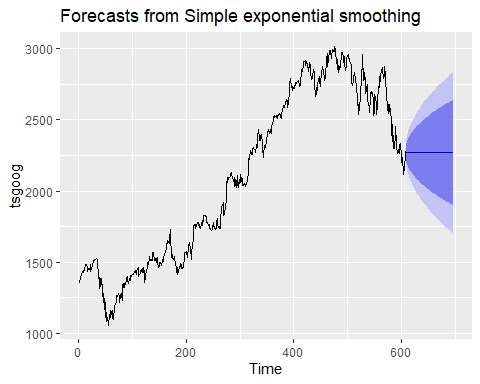
g1<-accuracy(ses.goog1)  
#alpha=0.35  
ses.goog2 <- ses(tsgoog, alpha = 0.35,h=90)  
autoplot(ses.goog1)



g2<-accuracy(ses.goog1)  
#alpha=0.55  
ses.goog3 <- ses(tsgoog, alpha = 0.55,h=90)  
autoplot(ses.goog3)



g3<-accuracy(ses.goog2)  
#alpha=0.75  
ses.goog4 <- ses(tsgoog, alpha = 0.75,h=90)  
autoplot(ses.goog4)



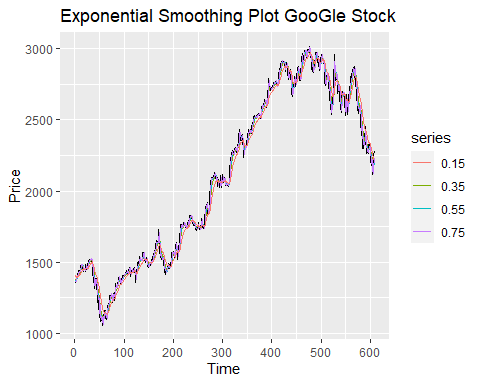
g4<-accuracy(ses.goog4)

#Compare Forecast with Actual time series.

##Accuracy Table  
g<-rbind(g1,g2,g3,g4)  
rownames(g)<-c("Alpha=0.15","Alpha=0.35","Alpha=0.55","Alpha=0.75")  
g

## ME RMSE MAE MPE MAPE MASE ACF1  
## Alpha=0.15 9.060792 69.69161 54.29045 0.38717782 2.734334 1.876124 0.8161944  
## Alpha=0.35 9.060792 69.69161 54.29045 0.38717782 2.734334 1.876124 0.8161944  
## Alpha=0.55 3.969974 49.39431 36.91302 0.17364335 1.841889 1.275609 0.5872288  
## Alpha=0.75 1.974617 40.37962 29.43959 0.08464608 1.469905 1.017349 0.1662957

##Plot  
autoplot(ts(tsgoog))+autolayer(ses.goog2$fitted, series ="0.35")+autolayer(ses.goog3$fitted, series ="0.55")+autolayer(ses.goog4$fitted, series ="0.75")+autolayer(ses.goog1$fitted, series = "0.15")+ggtitle("Exponential Smoothing Plot GooGle Stock")+labs(x="Time", y="Price")

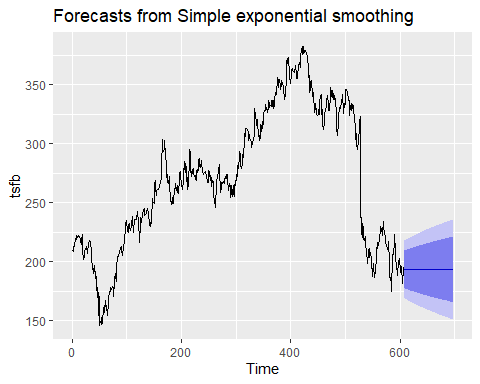


- Interpretation: - We can see the forecasting lines are flat which do not tell anything about the increasing trend as well as the seasonality of the time series

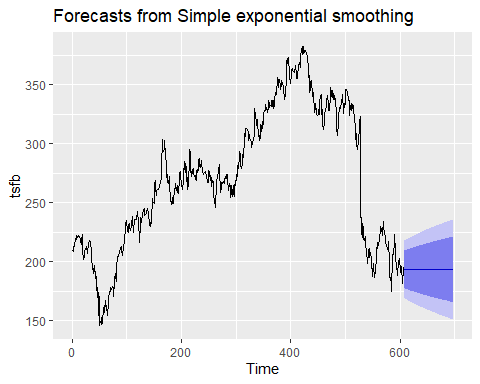
- According to the The accuracy tables, we can notice that the model with alpha=0.75 has the lowest error with all index (MSE, MAE, and MAPE). We conclude that alpha=0.75 gives out the best forecasting for all three companies. - Again in the plot, we can see that the Purple line presenting for the forecast with alpha =0.75 is the closest to the black line which is the actual data. In contrast, the farthest is the red line which is of the forecasting with alpha=0.15

### 3. FACEBOOK

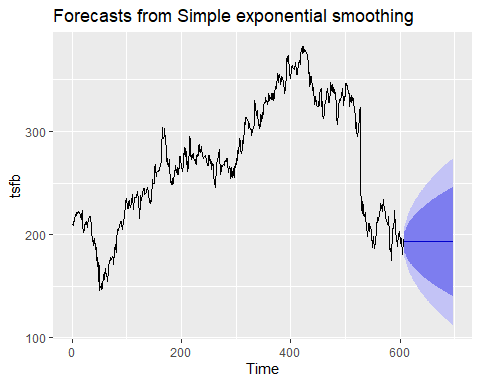
#alpha=0.15  
ses.fb1 <- ses(tsfb, alpha = 0.15,h=90)  
autoplot(ses.fb1)



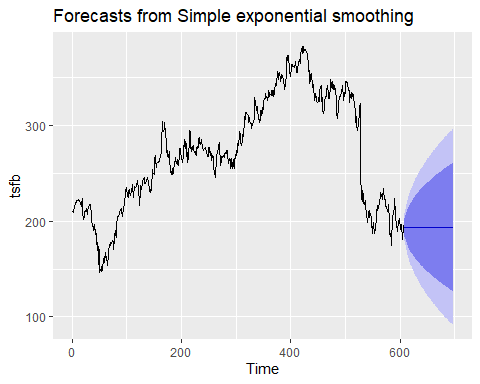
f1<-accuracy(ses.fb1)  
#alpha=0.35  
ses.fb2 <- ses(tsfb, alpha = 0.35,h=90)  
autoplot(ses.fb1)



f2<-accuracy(ses.fb1)  
#alpha=0.55  
ses.fb3 <- ses(tsfb, alpha = 0.55,h=90)  
autoplot(ses.fb3)



f3<-accuracy(ses.fb2)  
#alpha=0.75  
ses.fb4 <- ses(tsfb, alpha = 0.75,h=90)  
autoplot(ses.fb4)



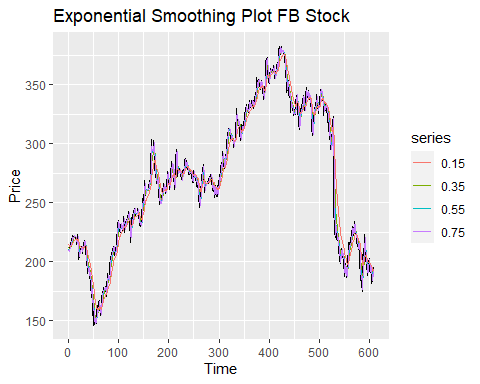
f4<-accuracy(ses.fb4)

#Compare Forecast with Actual time series.

##Accuracy Table  
f<-rbind(f1,f2,f3,f4)  
rownames(f)<-c("Alpha=0.15","Alpha=0.35","Alpha=0.55","Alpha=0.75")  
f

## ME RMSE MAE MPE MAPE MASE  
## Alpha=0.15 -0.23476304 12.449337 9.040050 -0.35875176 3.646006 1.856373  
## Alpha=0.35 -0.23476304 12.449337 9.040050 -0.35875176 3.646006 1.856373  
## Alpha=0.55 -0.09202154 8.921839 6.389168 -0.15283637 2.536159 1.312015  
## Alpha=0.75 -0.03530615 7.317282 4.995546 -0.07215702 1.974010 1.025835  
## ACF1  
## Alpha=0.15 0.8166149  
## Alpha=0.35 0.8166149  
## Alpha=0.55 0.5939729  
## Alpha=0.75 0.1918779

##Plot  
autoplot(ts(tsfb))+autolayer(ses.fb2$fitted, series ="0.35")+autolayer(ses.fb3$fitted, series ="0.55")+autolayer(ses.fb4$fitted, series ="0.75")+autolayer(ses.fb1$fitted, series = "0.15")+ggtitle("Exponential Smoothing Plot FB Stock")+labs(x="Time", y="Price")



Simple Exponential Smoothing: We can see the forecasting lines are flat which does not tell anything about the increasing trend as well as the seasonality of the time series

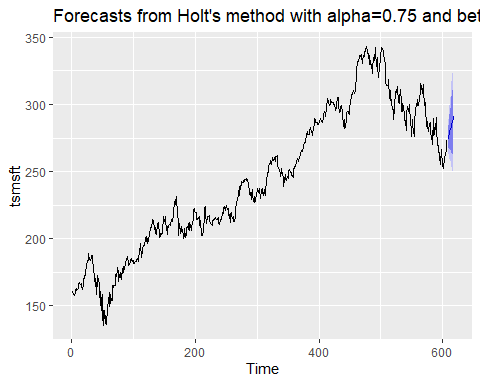
According to the accuracy tables, we can notice that the model with alpha=0.75 has the lowest error with all index (MSE, MAE, and MAPE). We conclude that alpha=0.75 gives out the best forecasting for all three companies. Again in the plot, we can see that the Purple line presenting for the forecast with alpha =0.75 is the closest to the black line which is the actual data. In contrast, the farthest is the red line which is of the forecasting with alpha=0.15

## Double Exponential Smoothing.

### MSFT

#Plot

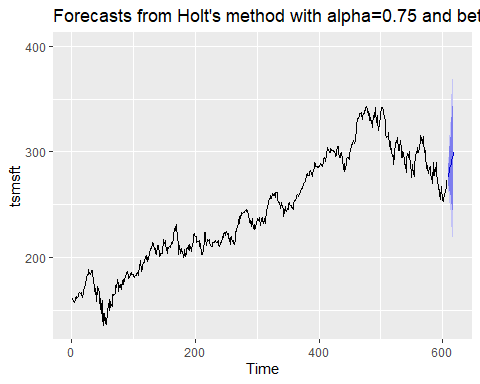
##Double Exponential Smoothing  
#alpha=0.75, beta=0.15  
holt.msft1<-holt(tsmsft,h=10,alpha=0.75, beta=0.15)  
autoplot(holt.msft1)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.15")



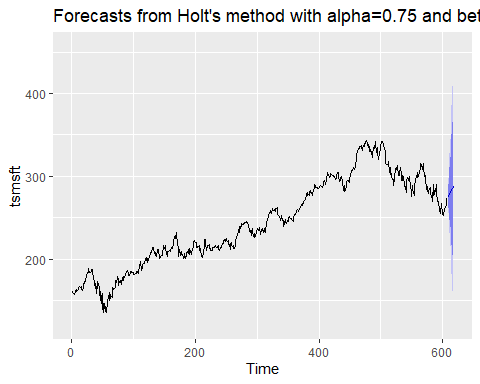
mh1<-accuracy(holt.msft1)  
summary(holt.msft1)

##   
## Forecast method: Holt's method  
##   
## Model Information:  
## Holt's method   
##   
## Call:  
## holt(y = tsmsft, h = 10, alpha = 0.75, beta = 0.15)   
##   
## Smoothing parameters:  
## alpha = 0.75   
## beta = 0.15   
##   
## Initial states:  
## l = 159.7214   
## b = 0.1848   
##   
## sigma: 4.8282  
##   
## AIC AICc BIC   
## 5813.921 5813.960 5827.151   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 0.01816503 4.812242 3.578725 -0.002218948 1.534152 1.034003  
## ACF1  
## Training set -0.004774438  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 609 274.1183 267.9308 280.3059 264.6553 283.5813  
## 610 275.9598 267.6353 284.2842 263.2286 288.6910  
## 611 277.8012 267.2416 288.3609 261.6516 293.9508  
## 612 279.6427 266.7339 292.5515 259.9004 299.3850  
## 613 281.4841 266.1084 296.8599 257.9690 304.9993  
## 614 283.3256 265.3658 301.2854 255.8584 310.7928  
## 615 285.1670 264.5082 305.8259 253.5721 316.7620  
## 616 287.0085 263.5387 310.4783 251.1145 322.9025  
## 617 288.8500 262.4603 315.2396 248.4905 329.2094  
## 618 290.6914 261.2763 320.1066 245.7048 335.6780

#alpha=0.75, beta=0.45  
holt.msft2<-holt(tsmsft,h=10,alpha=0.75, beta=0.45)  
autoplot(holt.msft2)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.45")



mh2<-accuracy(holt.msft2)  
#alpha=0.75, beta=0.75  
holt.msft3<-holt(tsmsft,h=10,alpha=0.75, beta=0.75)  
autoplot(holt.msft3)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.75")



mh3<-accuracy(holt.msft3)

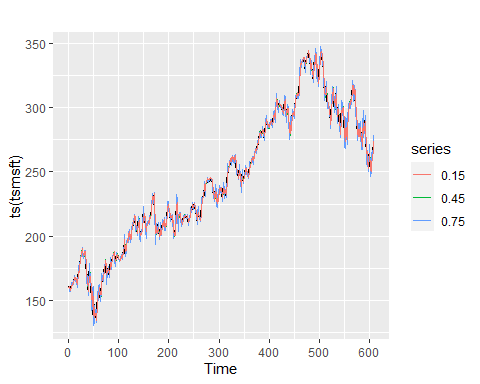
Looking at the forecasting lines, we can tell that there is an increase in the trend of the stock price, which is partly show the actual pattern of the real time series. We are still looking for the seasonal factor.

#Compare The actual with Forecasting Time Series

## Accuracy table  
mh<-rbind(mh1, mh2, mh3)  
rownames(mh)<-c("alpha=0.75, beta=0.15", "alpha=0.75, beta=0.45", "alpha=0.75, beta=0.75")  
mh

## ME RMSE MAE MPE MAPE  
## alpha=0.75, beta=0.15 0.01816503 4.812242 3.578725 -0.002218948 1.534152  
## alpha=0.75, beta=0.45 0.01301619 5.242568 3.902883 -0.004125030 1.680207  
## alpha=0.75, beta=0.75 0.00691913 5.811582 4.344308 -0.007993810 1.876397  
## MASE ACF1  
## alpha=0.75, beta=0.15 1.034003 -0.004774438  
## alpha=0.75, beta=0.45 1.127663 -0.144672539  
## alpha=0.75, beta=0.75 1.255204 -0.277470790

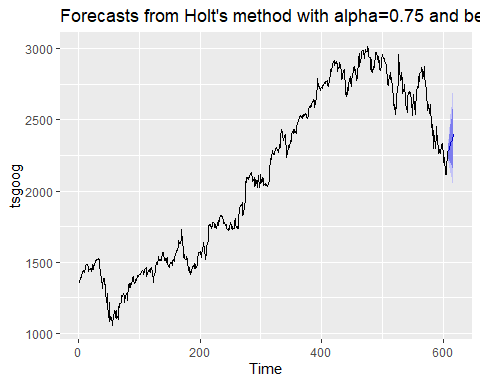
## Plot  
autoplot(ts(tsmsft))+autolayer(holt.msft2$fitted, series ="0.45")+autolayer(holt.msft3$fitted, series ="0.75")+autolayer(holt.msft1$fitted, series = "0.15")



According to the accuracy table which compare the metrics of 3 models, we can see that model with alpha=0.75 and beta=0.15 yields the lowest error. We decided that this on generate the best forecasting. Again in this plot, we can see that the Blue line presenting for the forecast with beta =0.15 is the closest to the black line which is the actual data. In contrast, the farthest is the red line which is of the forecasting with beta=0.75

### GOOG

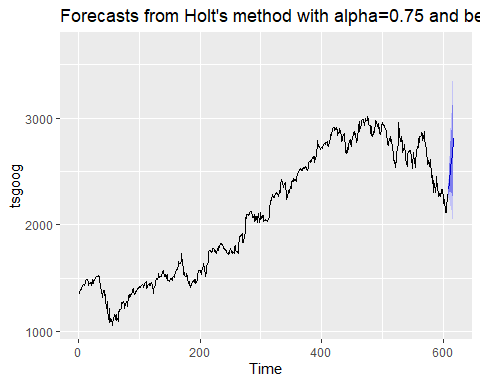
##Double Exponential Smoothing  
#alpha=0.75, beta=0.15  
holt.goog1<-holt(tsgoog,h=10,alpha=0.75, beta=0.15)  
autoplot(holt.goog1)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.15")



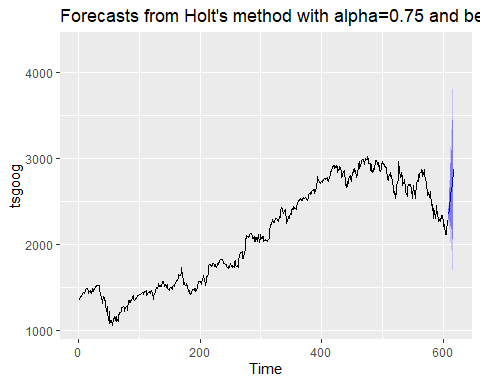
gh1<-accuracy(holt.goog1)  
summary(holt.goog1)

##   
## Forecast method: Holt's method  
##   
## Model Information:  
## Holt's method   
##   
## Call:  
## holt(y = tsgoog, h = 10, alpha = 0.75, beta = 0.15)   
##   
## Smoothing parameters:  
## alpha = 0.75   
## beta = 0.15   
##   
## Initial states:  
## l = 1355.4204   
## b = 9.137   
##   
## sigma: 42.1254  
##   
## AIC AICc BIC   
## 8448.005 8448.045 8461.236   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 0.04010488 41.98663 30.58763 -0.004533163 1.520202 1.057022  
## ACF1  
## Training set 0.1031728  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 609 2281.141 2227.155 2335.126 2198.576 2363.705  
## 610 2293.935 2221.304 2366.566 2182.856 2405.014  
## 611 2306.730 2214.597 2398.862 2165.825 2447.634  
## 612 2319.524 2206.895 2432.153 2147.273 2491.775  
## 613 2332.319 2198.166 2466.471 2127.150 2537.487  
## 614 2345.113 2188.415 2501.812 2105.463 2584.763  
## 615 2357.908 2177.660 2538.155 2082.243 2633.573  
## 616 2370.702 2165.929 2575.476 2057.529 2683.876  
## 617 2383.497 2153.248 2613.745 2031.362 2735.632  
## 618 2396.291 2139.645 2652.938 2003.785 2788.798

#alpha=0.75, beta=0.45  
holt.goog2<-holt(tsgoog,h=10,alpha=0.75, beta=0.45)  
autoplot(holt.goog2)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.45")



gh2<-accuracy(holt.goog2)  
#alpha=0.75, beta=0.75  
holt.goog3<-holt(tsgoog,h=10,alpha=0.75, beta=0.75)  
autoplot(holt.goog3)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.75")

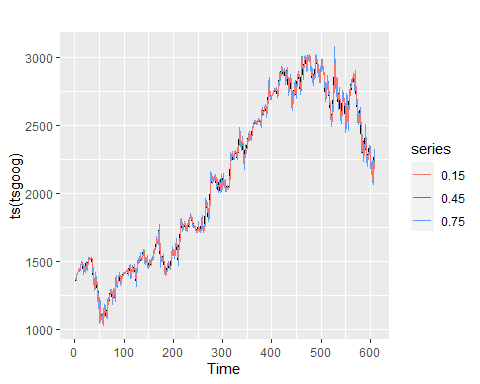


gh3<-accuracy(holt.goog3)

## Accuracy table  
gh<-rbind(gh1, gh2, gh3)  
rownames(gh)<-c("alpha=0.75, beta=0.15", "alpha=0.75, beta=0.45", "alpha=0.75, beta=0.75")  
gh

## ME RMSE MAE MPE MAPE  
## alpha=0.75, beta=0.15 0.04010488 41.98663 30.58763 -0.0045331635 1.520202  
## alpha=0.75, beta=0.45 0.15783671 45.39122 33.50501 0.0006246619 1.674992  
## alpha=0.75, beta=0.75 0.11298322 49.37153 36.81460 -0.0015111352 1.842899  
## MASE ACF1  
## alpha=0.75, beta=0.15 1.057022 0.10317277  
## alpha=0.75, beta=0.45 1.157838 -0.03491488  
## alpha=0.75, beta=0.75 1.272208 -0.18590664

## Plot  
autoplot(ts(tsgoog))+autolayer(holt.goog2$fitted, series ="0.45")+autolayer(holt.goog3$fitted, series ="0.75")+autolayer(holt.goog1$fitted, series = "0.15")

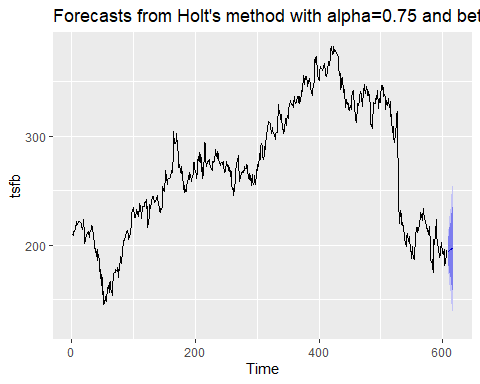


Looking at the forecasting lines, we can tell that there is an increase in the trend of the stock price, which is partly show the actual pattern of the real time series. We are still looking for the seasonal factor.

According to the accuracy table which compare the metrics of 3 models, we can see that model with alpha=0.75 and beta=0.15 yields the lowest error. We decided that this on generate the best forecasting. Again in this plot, we can see that the Blue line presenting for the forecast with beta =0.15 is the closest to the black line which is the actual data. In contrast, the farthest is the red line which is of the forecasting with beta=0.75

## Facebook

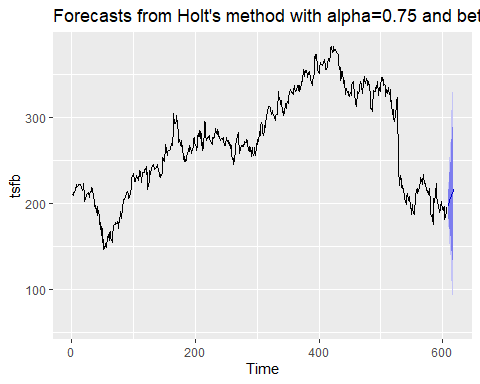
##Double Exponential Smoothing  
#alpha=0.75, beta=0.15  
holt.fb1<-holt(tsfb,h=10,alpha=0.75, beta=0.15)  
autoplot(holt.fb1)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.15")



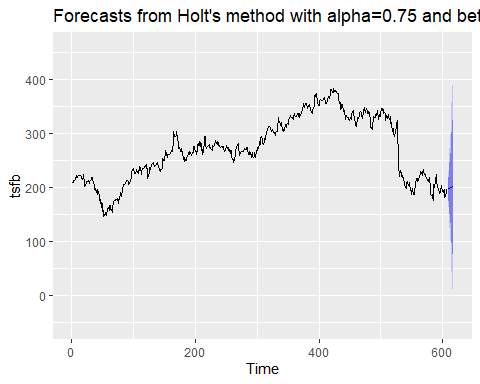
fh1<-accuracy(holt.fb1)  
summary(holt.fb1)

##   
## Forecast method: Holt's method  
##   
## Model Information:  
## Holt's method   
##   
## Call:  
## holt(y = tsfb, h = 10, alpha = 0.75, beta = 0.15)   
##   
## Smoothing parameters:  
## alpha = 0.75   
## beta = 0.15   
##   
## Initial states:  
## l = 208.1219   
## b = 1.1963   
##   
## sigma: 7.6766  
##   
## AIC AICc BIC   
## 6377.798 6377.838 6391.028   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set -0.008461136 7.651317 5.251206 -0.01239569 2.062707 1.078335  
## ACF1  
## Training set 0.1364871  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 609 194.1458 184.3078 203.9838 179.0999 209.1917  
## 610 194.5704 181.3348 207.8060 174.3282 214.8126  
## 611 194.9950 178.2055 211.7845 169.3176 220.6724  
## 612 195.4196 174.8950 215.9442 164.0299 226.8093  
## 613 195.8442 171.3973 220.2911 158.4559 233.2325  
## 614 196.2688 167.7132 224.8243 152.5968 239.9407  
## 615 196.6934 163.8465 229.5403 146.4584 246.9284  
## 616 197.1180 159.8017 234.4343 140.0477 254.1883  
## 617 197.5426 155.5839 239.5013 133.3723 261.7129  
## 618 197.9672 151.1980 244.7364 126.4399 269.4945

#alpha=0.75, beta=0.45  
holt.fb2<-holt(tsfb,h=10,alpha=0.75, beta=0.45)  
autoplot(holt.fb2)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.45")



fh2<-accuracy(holt.fb2)  
#alpha=0.75, beta=0.75  
holt.fb3<-holt(tsfb,h=10,alpha=0.75, beta=0.75)  
autoplot(holt.fb3)+ggtitle("Forecasts from Holt's method with alpha=0.75 and beta=0.75")

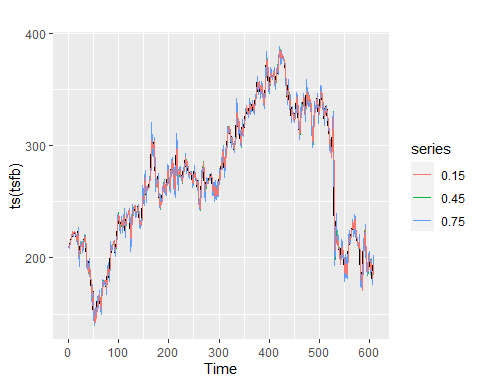


fh3<-accuracy(holt.fb3)

## Accuracy table  
fh<-rbind(fh1, fh2, fh3)  
rownames(fh)<-c("alpha=0.75, beta=0.15", "alpha=0.75, beta=0.45", "alpha=0.75, beta=0.75")  
fh

## ME RMSE MAE MPE MAPE  
## alpha=0.75, beta=0.15 -0.0084611355 7.651317 5.251206 -0.012395688 2.062707  
## alpha=0.75, beta=0.45 0.0037329325 8.268493 5.761188 -0.004802782 2.268866  
## alpha=0.75, beta=0.75 0.0007845596 8.915720 6.333224 -0.005798596 2.504486  
## MASE ACF1  
## alpha=0.75, beta=0.15 1.078335 0.136487130  
## alpha=0.75, beta=0.45 1.183059 0.003431872  
## alpha=0.75, beta=0.75 1.300527 -0.154636030

## Plot  
autoplot(ts(tsfb))+autolayer(holt.fb2$fitted, series ="0.45")+autolayer(holt.fb3$fitted, series ="0.75")+autolayer(holt.fb1$fitted, series = "0.15")



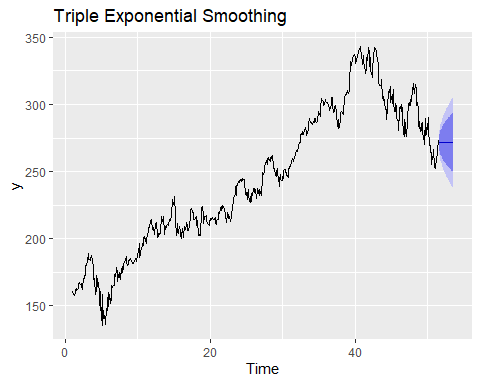
Looking at the forecasting lines, we can tell that there is an increase in the trend of the stock price, which is partly show the actual pattern of the real time series. We are still looking for the seasonal factor.

According to the accuracy table which compare the metrics of 3 models, we can see that model with alpha=0.75 and beta=0.15 yields the lowest error. We decided that this on generate the best forecasting. Again in this plot, we can see that the Blue line presenting for the forecast with beta =0.15 is the closest to the black line which is the actual data. In contrast, the farthest is the red line which is of the forecasting with beta=0.75

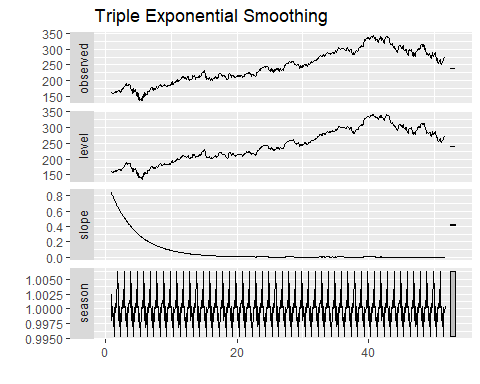
## V. Triple Exponential Smoothing.

### 1. MSFT

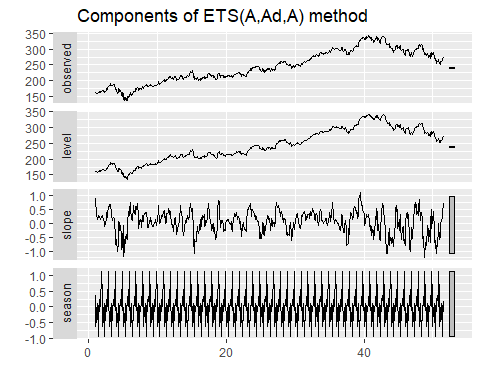
##Automatically selected   
y<-ts(tsmsft, frequency = 12)  
ets.msft1<-ets(y,model="ZZZ", alpha=0.75)  
fc1<-forecast(ets.msft1)  
autoplot(forecast(ets.msft1), h=5)+ggtitle("Triple Exponential Smoothing")



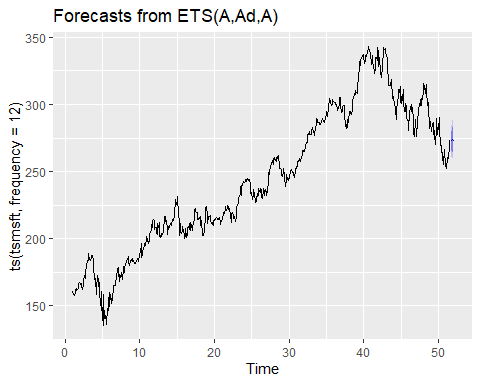
tm1<-accuracy(fc1)  
  
##Multiplicative  
ets.msft2<-ets(ts(tsmsft, frequency = 12),model="MAM",alpha=0.75)  
autoplot(ets.msft2)+ggtitle("Triple Exponential Smoothing")



fc2<-forecast(ets.msft2,h=5)  
tm2<-accuracy(fc2)  
  
  
### Additive  
ets.msft3<-ets(ts(tsmsft, frequency = 12),model="AAA",alpha=0.75)  
autoplot(ets.msft3)



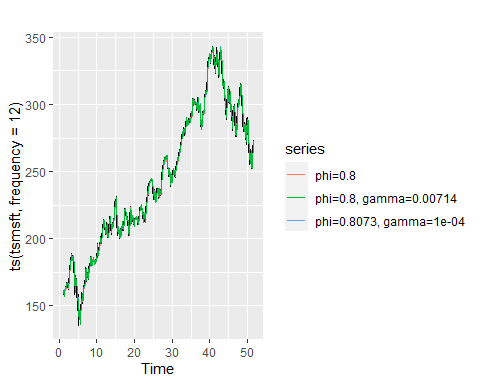
fc3<-forecast(ets.msft3,h=5)  
autoplot(fc3)



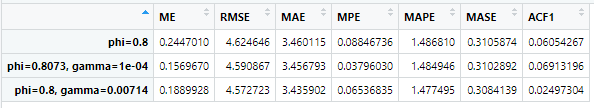
tm3<-accuracy(fc3)

##Compare forecasting and actual time series

#Accuracy table  
tm<-rbind(tm1,tm2,tm3)  
rownames(tm)<-c("phi=0.8","phi=0.8073, gamma=1e-04","phi=0.8, gamma=0.00714")  
  
  
#Plot  
autoplot(ts(tsmsft, frequency = 12))+autolayer(fc1$fitted, series ="phi=0.8")+autolayer(fc2$fitted, series ="phi=0.8073, gamma=1e-04")+autolayer(fc3$fitted, series = "phi=0.8, gamma=0.00714")



#Accuracy table

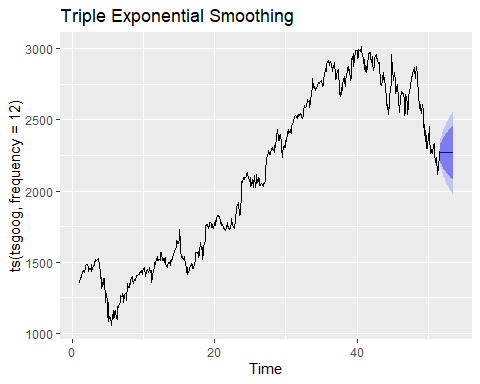


As we can see there is not much difference in the error of different models. The errors fluctuate within 0.2 unit values.

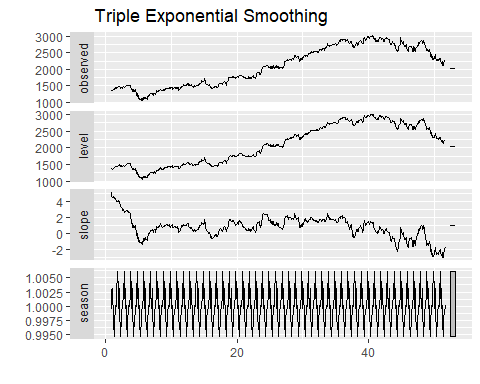
And the model with phi=0.8, gamma=0.00714, which is of Additive method, yields the best result.

### GOOGLE

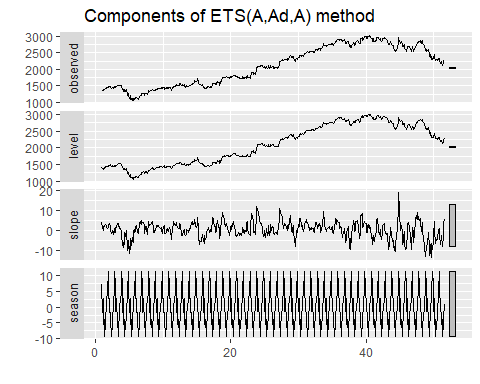
##Automatically selected   
  
ets.goog1<-ets(ts(tsgoog, frequency = 12),model="ZZZ", alpha=0.75)  
gfc1<-forecast(ets.goog1)  
autoplot(forecast(ets.goog1), h=5)+ggtitle("Triple Exponential Smoothing")



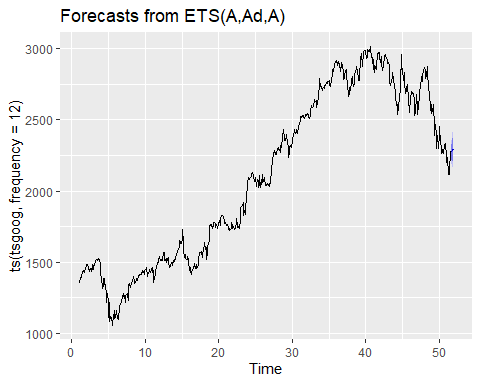
tg1<-accuracy(gfc1)  
  
##Multiplicative  
ets.goog2<-ets(ts(tsgoog, frequency = 12),model="MAM",alpha=0.75)  
autoplot(ets.goog2)+ggtitle("Triple Exponential Smoothing")



gfc2<-forecast(ets.goog2,h=5)  
tg2<-accuracy(gfc2)  
  
  
### Additive  
ets.goog3<-ets(ts(tsgoog, frequency = 12),model="AAA",alpha=0.75)  
autoplot(ets.goog3)

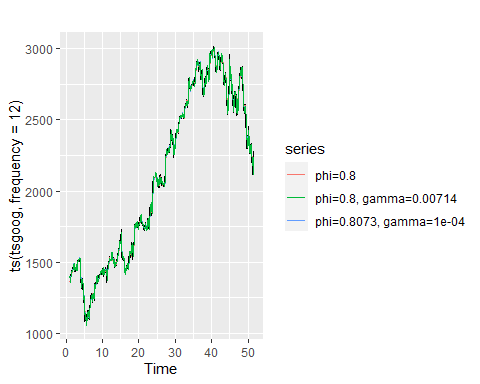


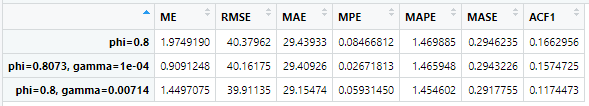
gfc3<-forecast(ets.goog3,h=5)  
autoplot(gfc3)



tg3<-accuracy(gfc3)

#Accuracy table  
tg<-rbind(tg1,tg2,tg3)  
rownames(tg)<-c("phi=0.8","phi=0.8073, gamma=1e-04","phi=0.8, gamma=0.00714")  
  
  
#Plot  
autoplot(ts(tsgoog, frequency = 12))+autolayer(gfc1$fitted, series ="phi=0.8")+autolayer(gfc2$fitted, series ="phi=0.8073, gamma=1e-04")+autolayer(gfc3$fitted, series = "phi=0.8, gamma=0.00714")

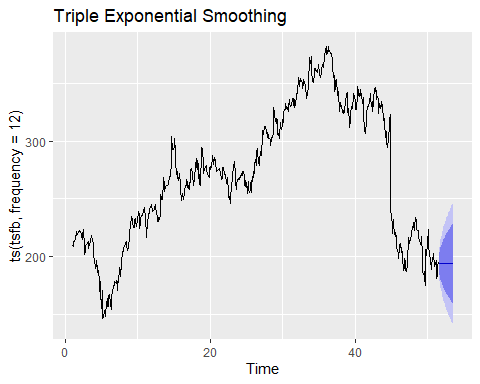




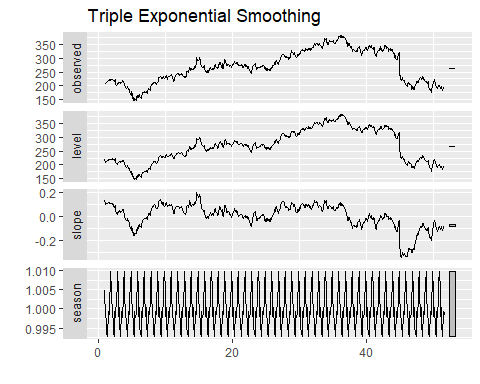
Again, there is not much difference in the error of 3 methods even though the method with phi=0.8 and gamma=0.00714 generates the best forecasting which is of additive method.

### Facebook

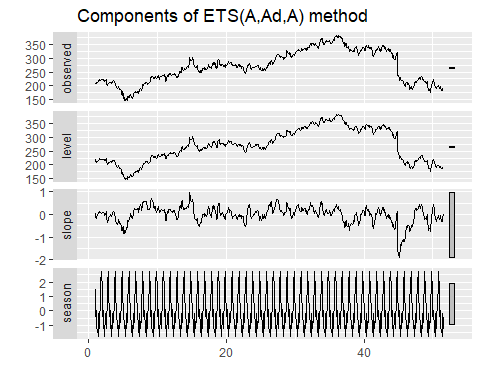
##Automatically selected   
  
ets.fb1<-ets(ts(tsfb, frequency = 12),model="ZZZ", alpha=0.75)  
ffc1<-forecast(ets.fb1)  
autoplot(forecast(ets.fb1), h=5)+ggtitle("Triple Exponential Smoothing")



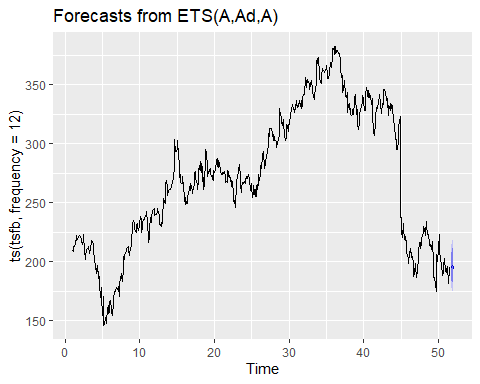
tf1<-accuracy(ffc1)  
  
##Multiplicative  
ets.fb2<-ets(ts(tsfb, frequency = 12),model="MAM",alpha=0.75)  
autoplot(ets.fb2)+ggtitle("Triple Exponential Smoothing")



ffc2<-forecast(ets.fb2,h=5)  
tf2<-accuracy(ffc2)  
  
  
### Additive  
ets.fb3<-ets(ts(tsfb, frequency = 12),model="AAA",alpha=0.75)  
autoplot(ets.fb3)

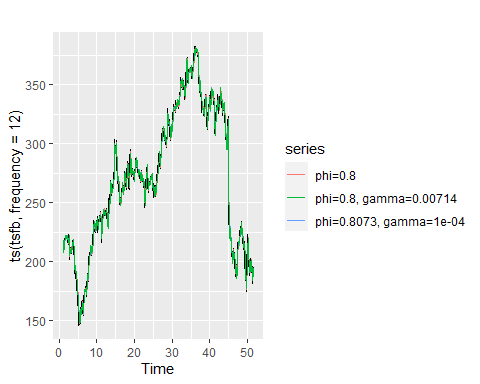


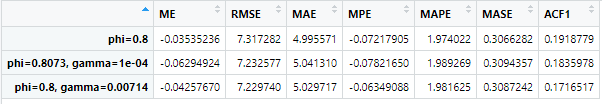
ffc3<-forecast(ets.fb3,h=5)  
autoplot(ffc3)



tf3<-accuracy(ffc3)

#Accuracy table  
tf<-rbind(tf1,tf2,tf3)  
rownames(tf)<-c("phi=0.8","phi=0.8073, gamma=1e-04","phi=0.8, gamma=0.00714")  
  
  
#Plot  
autoplot(ts(tsfb, frequency = 12))+autolayer(ffc1$fitted, series ="phi=0.8")+autolayer(ffc2$fitted, series ="phi=0.8073, gamma=1e-04")+autolayer(ffc3$fitted, series = "phi=0.8, gamma=0.00714")





Similar to these two previous companies, additive method produces the best forecasting compare two the other two.

1. **Compare the accuracy of Simple, Triple, and Double**

#MSFT  
compare.msft<-rbind(m4, mh1,tm3)  
rownames(compare.msft)<-c("Simple", "Double","Triple")  
compare.msft

## ME RMSE MAE MPE MAPE MASE  
## Simple 0.24463211 4.624645 3.460080 0.088424355 1.486788 0.9997233  
## Double 0.01816503 4.812242 3.578725 -0.002218948 1.534152 1.0340033  
## Triple 0.18899284 4.572723 3.435902 0.065368349 1.477495 0.3084139  
## ACF1  
## Simple 0.060548243  
## Double -0.004774438  
## Triple 0.024973040

#Google  
compare.goog<-rbind(g4, gh1,tg3)  
rownames(compare.goog)<-c("Simple", "Double","Triple")  
compare.goog

## ME RMSE MAE MPE MAPE MASE ACF1  
## Simple 1.97461748 40.37962 29.43959 0.084646079 1.469905 1.0173489 0.1662957  
## Double 0.04010488 41.98663 30.58763 -0.004533163 1.520202 1.0570220 0.1031728  
## Triple 1.44970754 39.91135 29.15474 0.059314497 1.454602 0.2917755 0.1174473

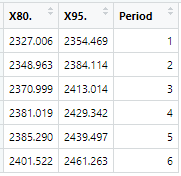
#Facebook  
  
compare.fb<-rbind(f4, fh1,tf3)  
rownames(compare.fb)<-c("Simple", "Double","Triple")  
compare.fb

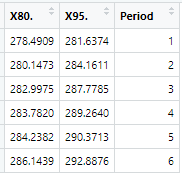
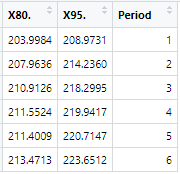
## ME RMSE MAE MPE MAPE MASE ACF1  
## Simple -0.035306152 7.317282 4.995546 -0.07215702 1.974010 1.0258349 0.1918779  
## Double -0.008461136 7.651317 5.251206 -0.01239569 2.062707 1.0783348 0.1364871  
## Triple -0.042576704 7.229740 5.029717 -0.06349088 1.981625 0.3087242 0.1716517

From the Accuracy table of three companies, we can easily notice that the triple exponential smoothing with alpha = 0.75, beta = 0.0611, phi=0.8, gamma=0.00714 generates the best forecasting.

## Forecasting Stock Price in Short Term (6months) and Long Term (2 years)

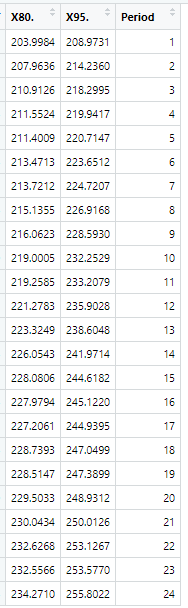
1. Short-term

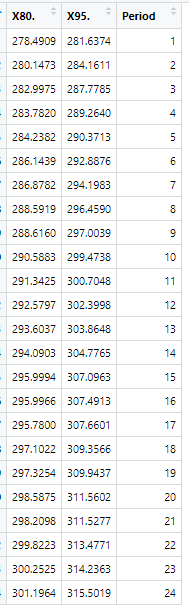
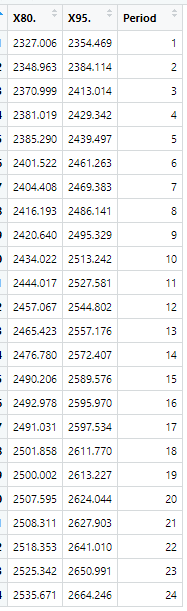
Facebook: table z1 Google: table w1 Microsoft: table t1



According to this prediction, I will choose Facebook to invest in short-term because its price increases the fastest.

1. Long-term

Facebook: table z Google: table w Microsoft: table t



According to the 6-month prediction, I will invest in Google company because its stock price increases the fastest which is around 300$ within 2 years. While stock prices of Facebook and Microsoft increase around 50 and 100 dollars respectively.

## Regression model

1. **Microsoft**

df1<-data.frame(tsmsft)  
df1$Period<-c(1:608)   
#Check Na Values.   
sum(is.na(tsmsft))

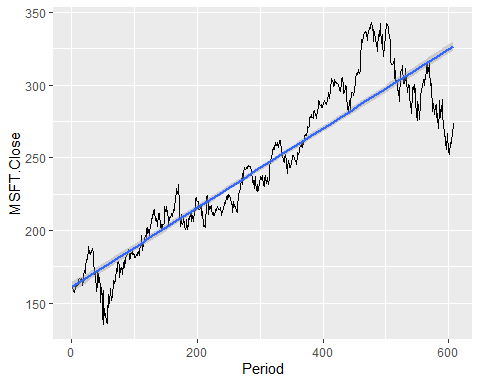
## [1] 0

#Linear Regression model  
m.lm.model<-lm(MSFT.Close~Period, df1)  
summary(m.lm.model)

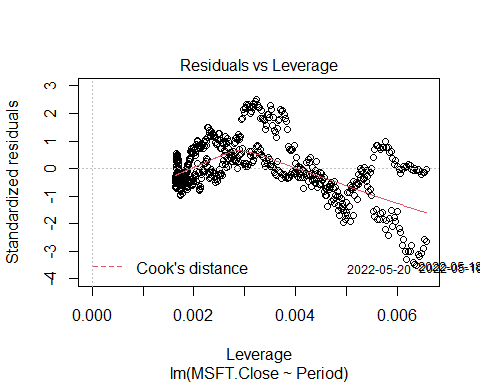
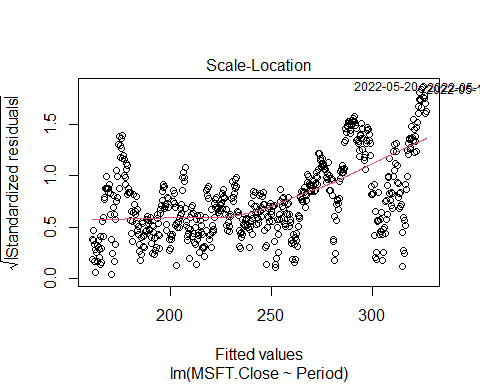
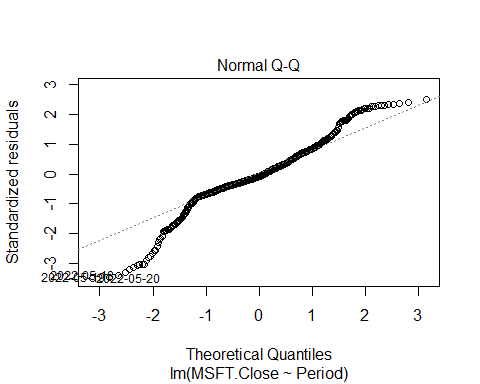
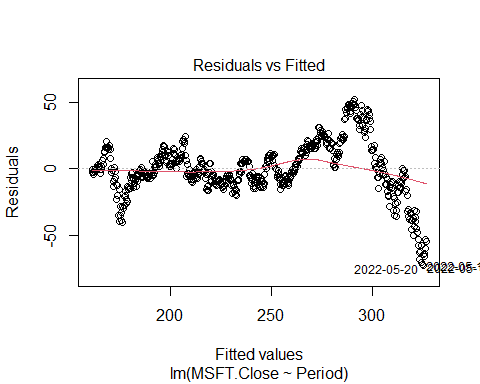
##   
## Call:  
## lm(formula = MSFT.Close ~ Period, data = df1)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -72.662 -9.675 -1.491 11.569 51.980   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.610e+02 1.690e+00 95.31 <2e-16 \*\*\*  
## Period 2.727e-01 4.807e-03 56.73 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 20.81 on 606 degrees of freedom  
## Multiple R-squared: 0.8415, Adjusted R-squared: 0.8413   
## F-statistic: 3218 on 1 and 606 DF, p-value: < 2.2e-16

#Plot actual time series and linear regession  
ggplot(df1, aes(x=Period, y=MSFT.Close))+geom\_line()+geom\_smooth(method = 'lm')

## `geom\_smooth()` using formula 'y ~ x'



##Linearity Assumtions  
plot(m.lm.model)



#Accuracy  
preds<-predict(m.lm.model, data=df$MSFT.Close)  
ml<-accuracy(m.lm.model)

1. **Google**

df2<-data.frame(tsgoog)  
df2$Period<-c(1:608)   
#Check Na Values.   
sum(is.na(tsgoog))

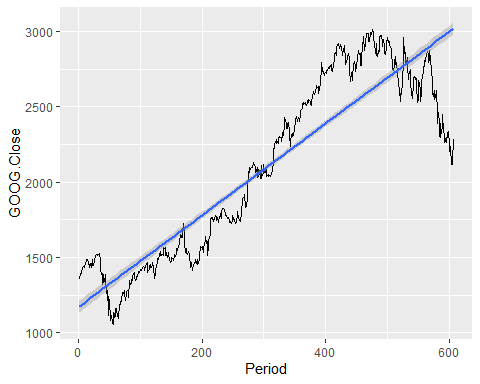
## [1] 0

#Linear Regression model  
g.lm.model<-lm(GOOG.Close~Period, df2)  
summary(g.lm.model)

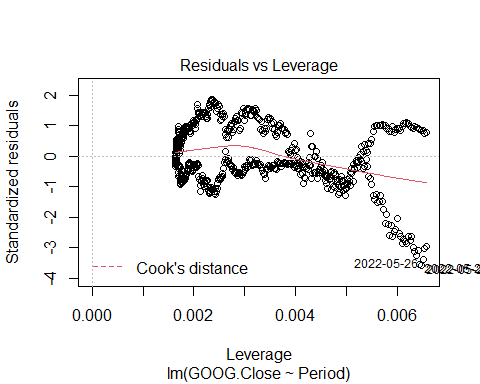
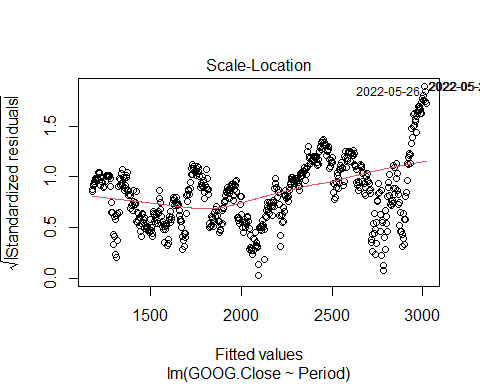
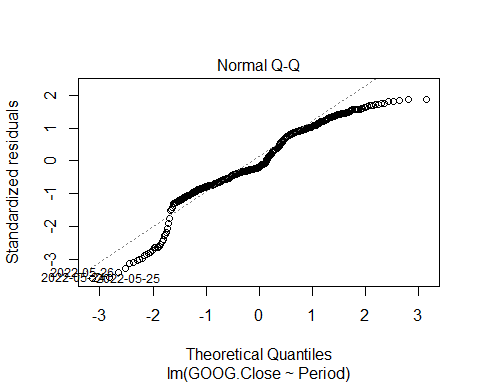
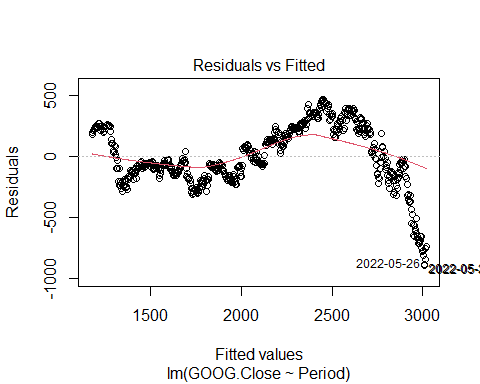
##   
## Call:  
## lm(formula = GOOG.Close ~ Period, data = df2)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -893.8 -144.9 -50.2 217.3 465.4   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.172e+03 2.036e+01 57.55 <2e-16 \*\*\*  
## Period 3.039e+00 5.793e-02 52.46 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 250.7 on 606 degrees of freedom  
## Multiple R-squared: 0.8195, Adjusted R-squared: 0.8193   
## F-statistic: 2752 on 1 and 606 DF, p-value: < 2.2e-16

#Plot actual time series and linear regession  
ggplot(df2, aes(x=Period, y=GOOG.Close))+geom\_line()+geom\_smooth(method = 'lm')

## `geom\_smooth()` using formula 'y ~ x'



##Linearity Assumtions  
plot(g.lm.model)



#Accuracy  
preds<-predict(g.lm.model, data=df$GOOG.Close)  
gl<-accuracy(g.lm.model)

1. **Facebook**

df3<-data.frame(tsfb)  
df3$Period<-c(1:608)   
#Check Na Values.   
sum(is.na(tsfb))

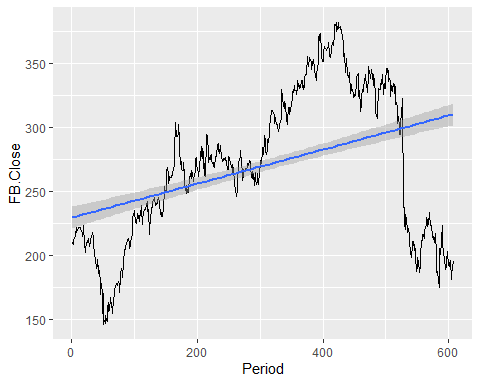
## [1] 0

#Linear Regression model  
f.lm.model<-lm(FB.Close~Period, df3)  
summary(f.lm.model)

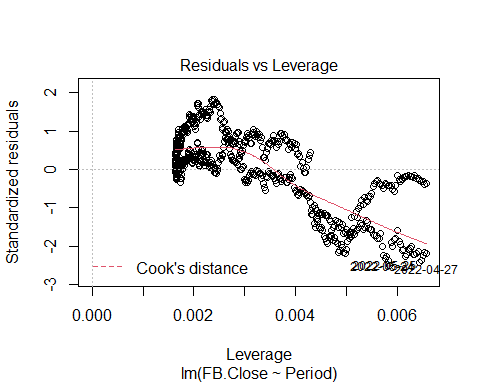
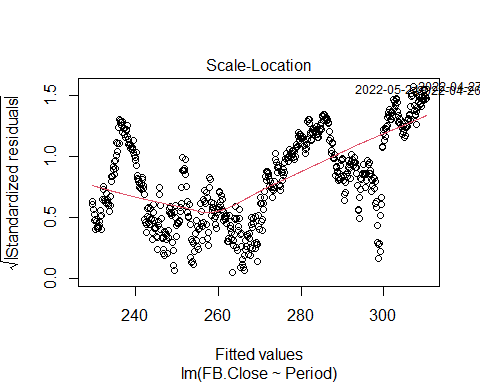
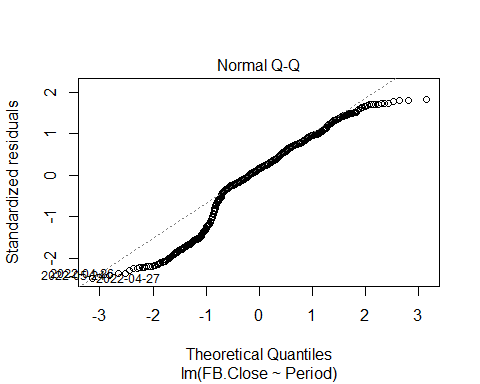
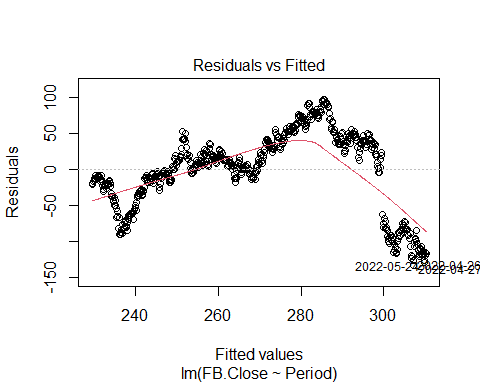
##   
## Call:  
## lm(formula = FB.Close ~ Period, data = df3)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -132.243 -22.005 8.285 38.631 96.672   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 229.37801 4.32824 53.0 <2e-16 \*\*\*  
## Period 0.13302 0.01231 10.8 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 53.3 on 606 degrees of freedom  
## Multiple R-squared: 0.1614, Adjusted R-squared: 0.1601   
## F-statistic: 116.7 on 1 and 606 DF, p-value: < 2.2e-16

#Plot actual time series and linear regession  
ggplot(df3, aes(x=Period, y=FB.Close))+geom\_line()+geom\_smooth(method = 'lm')

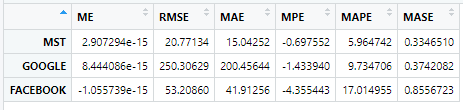
## `geom\_smooth()` using formula 'y ~ x'



#Linearity Assumtions  
plot(f.lm.model)



#Accuracy  
preds<-predict(f.lm.model, data=df$FB.Close)  
fl<-accuracy(f.lm.model)



The RMSE, MAPE and MAE of Linear Regression model is way higher than those of Exponential Smoothing Method.

1. **Linearity Assumption:**
2. Normality (Normal Q-Q)

Nearly half of the data are off the straight line, which showing that the data do not follow the normal distribution

1. Linearity (Residual vs Fitted)

We expect to see a random pattern of the dot; however, it is not the case here. So the linearity assumption is violated.

1. Homoscedasticity (Scale-Location)

If we have met the constant variance assumption, the points in the Scale-Location should be random band around the horizontal line. However, there is a distinct pattern.

1. Independence



P-value=0 so we reject the Null hypothesis and conclude that the residuals in this regression model are autocorrelated which violate the Linearity Assumption.

1. **Conclusion**

Triple Exponential Smoothing Method with alpha = 0.75, beta = 0.0611, phi=0.8, gamma=0.00714 generates the best forecasting.

I would invest in Facebook company for short-term (6 months) and Google for long-term (2 years) based on the forecasting.

**References**

<http://www.sthda.com/english/articles/39-regression-model-diagnostics/161-linear-regression-assumptions-and-diagnostics-in-r-essentials/?fbclid=IwAR0Kf3QxlGuQJHWT_m2Km_nLhdbsw4G_l2zHSN01v3JL4udGTF1PdDCh4vo>