Minitest1_Thunyaporn.R

CHATKEAWPAISAL

2021-02-11

```
setwd("/Users/CHATKEAWPAISAL/Desktop/4 term 2/EE435/assignment2_Thunyaporn")
cat(rep("\n",50))  #clear R Console

#install.packages("quantmod")
#install.packages("fBasics")
#install.packages("sn")
#install.packages("PerformanceAnalytics")
#install.packages("car")
#install.packages("tseries")
#install.packages("forecast")
library(quantmod)

## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

## Loading required package: TTR

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

library(fBasics)

## Loading required package: timeDate

## Loading required package: timeSeries

##
## Attaching package: 'timeSeries'

## The following object is masked from 'package:zoo':
##
##     time<-

##
## Attaching package: 'fBasics'
```

```
## The following object is masked from 'package:TTR':
##
##     volatility
```

```
library(sn)
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'sn'
```

```
## The following object is masked from 'package:fBasics':
##
##     vech
```

```
## The following object is masked from 'package:stats':
##
##     sd
```

```
library(PerformanceAnalytics)
```

```
##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness
```

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

```
library(car)
```

```
## Loading required package: carData
```

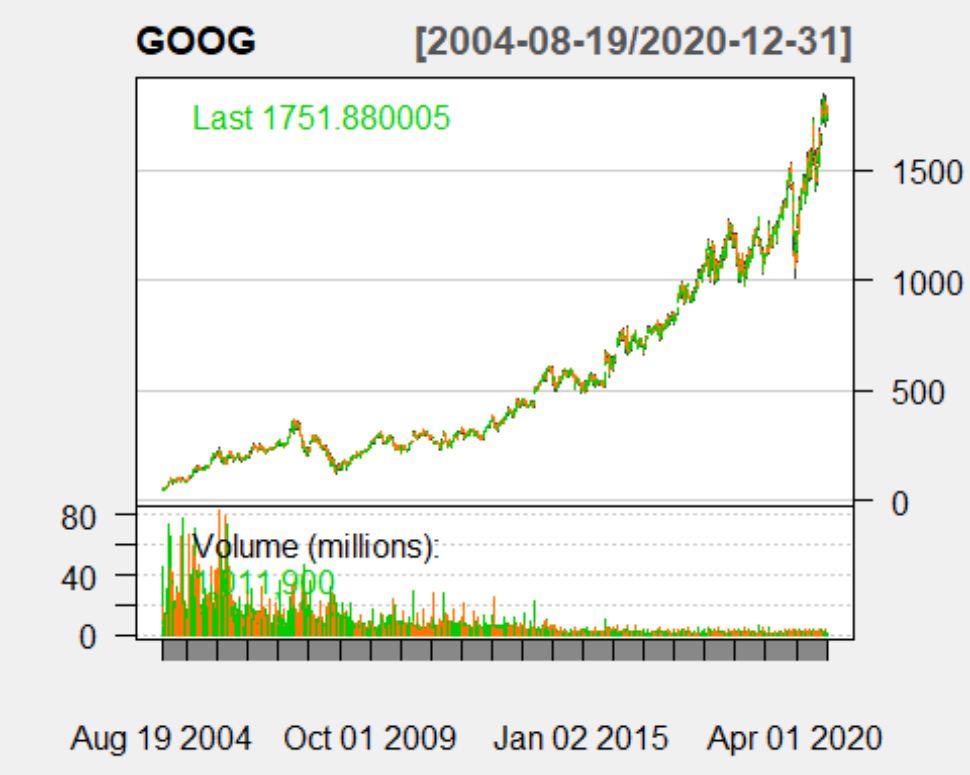
```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:fBasics':
##
##     densityPlot
```

```
library(tseries)
library(forecast)
getSymbols("GOOG", from="2004-08-19",to="2021-01-01")
```

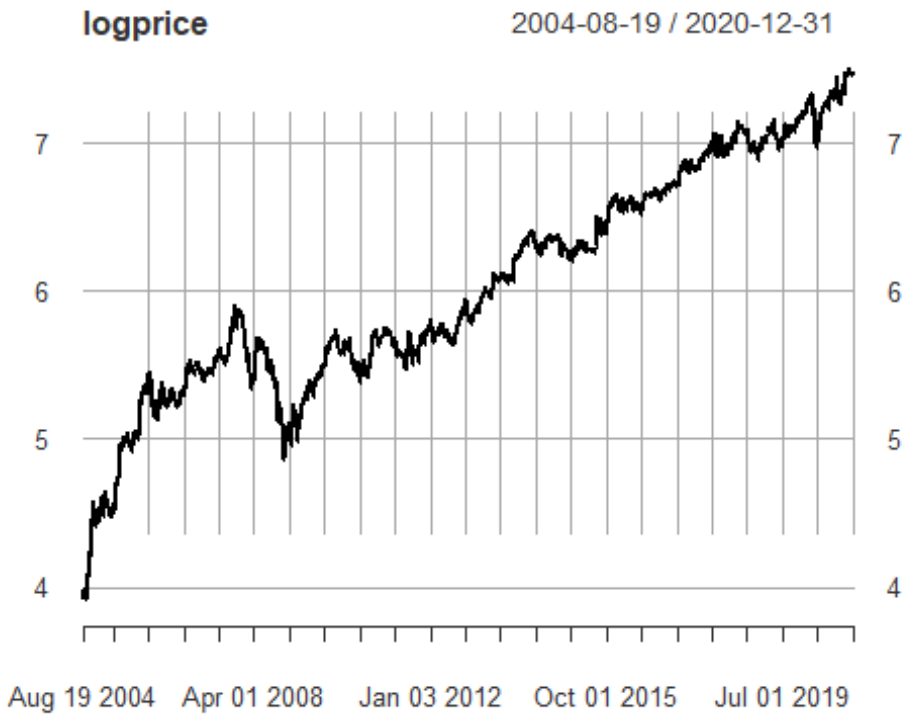
```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
```

```
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.

## [1] "GOOG"

dim(GOOG)

## [1] 4122    6

head(GOOG)

##            GOOG.Open GOOG.High GOOG.Low GOOG.Close GOOG.Volume
GOOG.Adjusted
## 2004-08-19  49.81329  51.83571 47.80083   49.98266    44871361
49.98266
## 2004-08-20  50.31640  54.33633 50.06235   53.95277    22942874
53.95277
## 2004-08-23  55.16822  56.52812 54.32139   54.49574    18342897
54.49574
## 2004-08-24  55.41230  55.59163 51.59162   52.23920    15319808
52.23920
## 2004-08-25  52.28403  53.79835 51.74604   52.80209     9232276
52.80209
## 2004-08-26  52.27905  53.77345 52.13459   53.75352     7128620
53.75352

tail(GOOG)

##            GOOG.Open GOOG.High GOOG.Low GOOG.Close GOOG.Volume
GOOG.Adjusted
## 2020-12-23  1728.110  1747.990 1725.040    1732.38     1033800
1732.38
## 2020-12-24  1735.000  1746.000 1729.110    1738.85      346800
1738.85
## 2020-12-28  1751.635  1790.728 1746.335    1776.09     1393000
1776.09
## 2020-12-29  1787.790  1792.440 1756.090    1758.72     1299400
1758.72
## 2020-12-30  1762.010  1765.095 1725.600    1739.52     1306100
1739.52
## 2020-12-31  1735.420  1758.930 1735.420    1751.88     1011900
1751.88

da= GOOG
chartSeries(GOOG,theme="white")
```

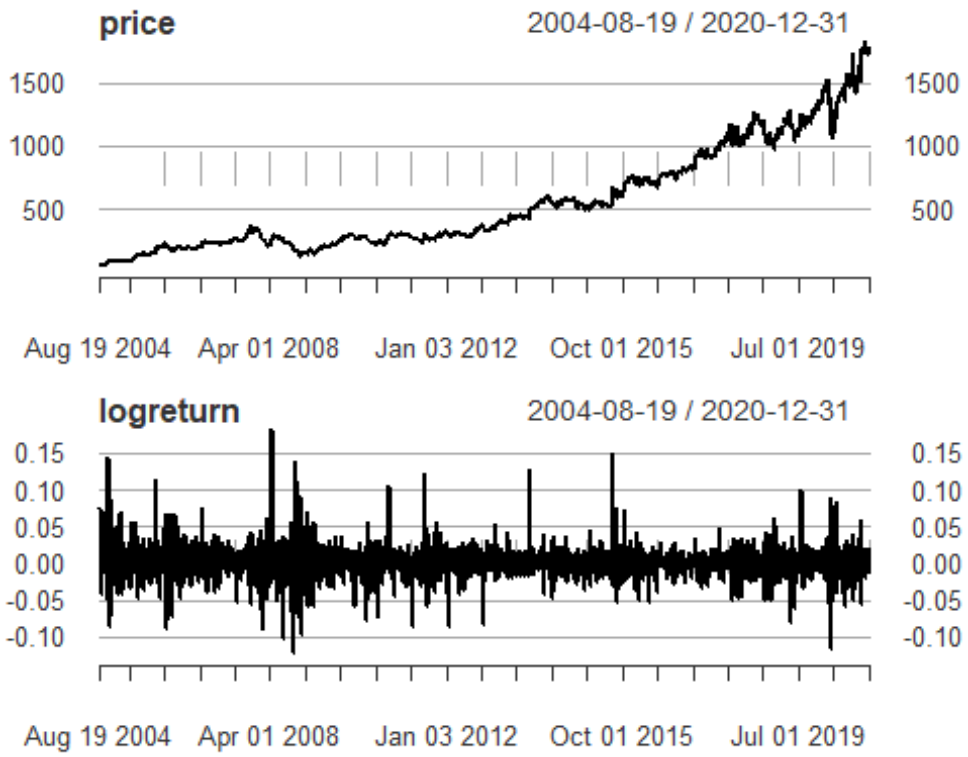



```
price=da[,6]
plot(price,type='l')
```

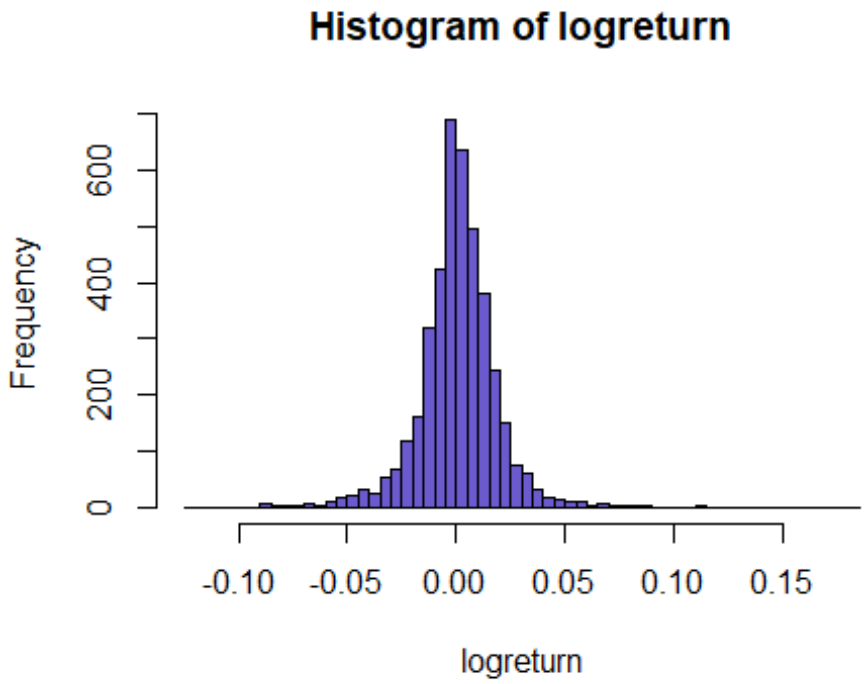
```
logprice=log(price)
plot(logprice,type='l')
```



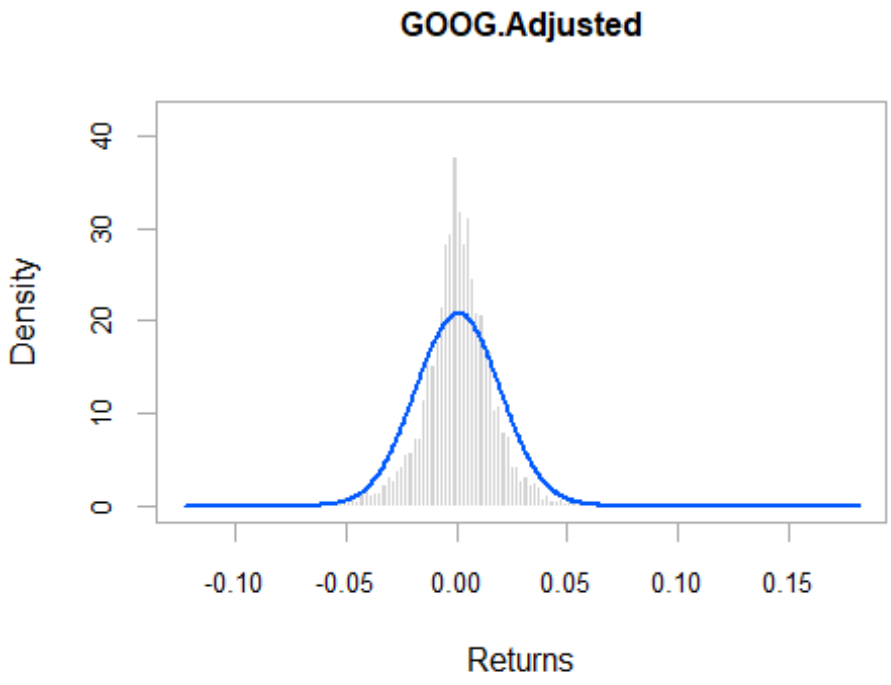

```
logreturn=diff(log(price))
simplereturn <-exp(logreturn)-1

#1a)
par(mfrow=c(2,1))
plot(price,type='l')
plot(logreturn, type='l')
```



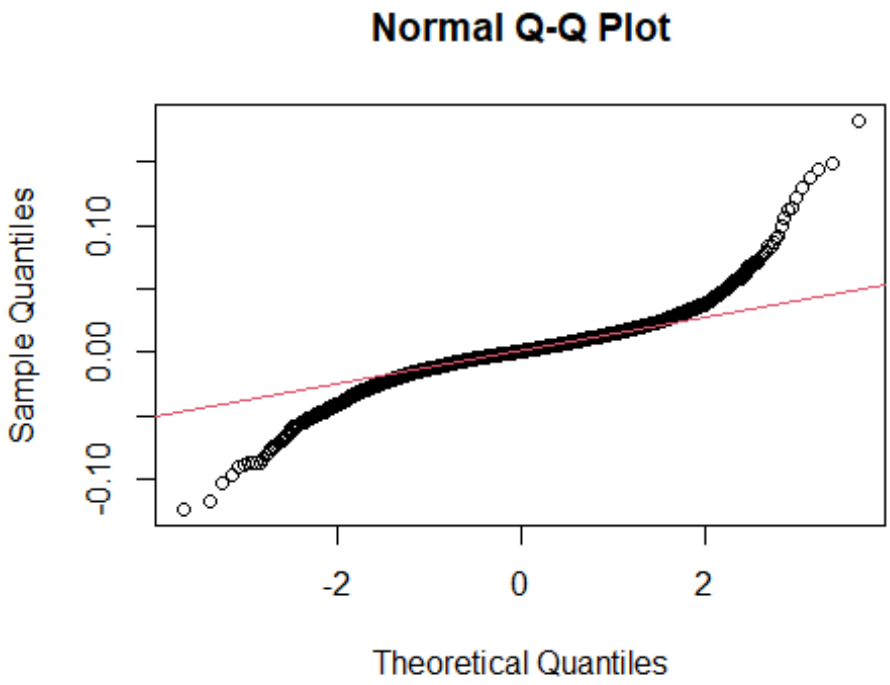

```
#1b)
table.Stats(logreturn)

##                  GOOG.Adjusted
## Observations         4121.0000
## NAs                     1.0000
## Minimum                -0.1234
## Quartile 1             -0.0074
## Median                  0.0007
## Arithmetic Mean         0.0009
## Geometric Mean          0.0007
## Quartile 3              0.0100
## Maximum                 0.1823
## SE Mean                 0.0003
## LCL Mean (0.95)         0.0003
## UCL Mean (0.95)         0.0014
## Variance                0.0004
## Stdev                   0.0191
## Skewness                0.4528
## Kurtosis                9.1391

#1c)
newlogreturn <- logreturn[2:nrow(logreturn)]
newsimplereturn <- simplereturn[2:nrow(logreturn)]

par(mfrow=c(1,1))
hist(logreturn, breaks = 100, col="slateblue")
```




```
chart.Histogram(logreturn, methods = c("add.normal"))
```



```
par(mfrow=c(1,1))
qqnorm((newlogreturn))
qqline(newlogreturn, col = 2)
```




```
jarque.bera.test(newlogreturn)
```

```
##
##  Jarque Bera Test
##
## data:  newlogreturn
## X-squared = 14482, df = 2, p-value < 2.2e-16
```

#1d)

H_0 : E(r) = 0.8

H_1 : E(r) != 0.8

t = 2.8957> 1.96, we reject the null hypothesis.

The return from GOOG does not have the mean log return of 0.8 at 95% confidence interval.

#1e)

```
t.test(newlogreturn)
```

```
## Warning in tstat + c(-cint, cint): Recycling array of length 1 in array-
vector arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## Warning in cint * stderr: Recycling array of length 1 in vector-array
arithmetic is deprecated.
##   Use c() or as.vector() instead.

##
##  One Sample t-test
##
## data:  newlogreturn
## t = 2.8957, df = 4120, p-value = 0.003803
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.0002787253 0.0014474425
## sample estimates:
##    mean of x
## 0.0008630839
```

```
#1f)
T=length(newlogreturn)
S = skewness(newlogreturn)
tst = S/sqrt(6/T)
tst
```

```
## [1] 11.8664
```

```
pv = 1-(pnorm(tst))
pv
```

```
## [1] 0
```

```
#1g)
k = kurtosis(newlogreturn)
tst = k/sqrt(24/T)
tst
```

```
## [1] 119.7564
```

```
pv = 2*(1-pnorm(tst))
pv
```

```
## [1] 0
```