# Time Series Exercise: Bitcoin Price

Bitcoin is a form of digital currency, created and held electronically. No one controls it. Bitcoins aren’t printed, like dollars or euros – they’re produced by lots of people running computers all around the world, using software that solves mathematical problems. It’s the first example of a growing category of money known as cryptocurrency (<https://bitcoin.org>).

Bitcoin daily price data was collected from 7/18/2010 to 3/9/2015 (<http://www.coindesk.com/price/>) and aggregate it into a monthly average Bitcoin price data “MonthlyBiPrice.csv”. The first column of the data is the month variable from July-2010 to March-2015, and the second column is monthly average price of Bitcoin. Please follow the instructions and answer the questions.

# 1. Import the monthly average Bitcoin price data into R. Check the summary of variable Monthly.Average.Price. What is the minimum average price of Bitcoin? What is the maximum average price of Bitcoin?

**Code**:

bit <- read.csv(file.choose(),stringsAsFactors = FALSE)

summary(bit)

str(bit)

**Ans**:

Month Monthly.Average.Price

Length:57 Min. : 0.0619

Class :character 1st Qu.: 4.9881

Mode :character Median : 14.0235

Mean :166.8490

3rd Qu.:274.3805

Max. :857.1822

#Min avg price = 0.0619

#Max avg price = 857.1822

ts()

# 2. Install the package "forecast" and create a time series object tprice. Select and

#only include the values from variable Monthly.

#Average.Price to construct this object. Set start month as c(2010, 7) and frequency = 12.

**Code**:

install.packages('forecast')

tprice <- ts(data = bit$Monthly.Average.Price, start = c(2010,7),frequency = 12)

**Ans**:

Jan Feb Mar Apr May Jun Jul Aug Sep Oct

2010 0.06528571 0.06454839 0.06188333 0.10877419

2011 0.36917419 0.92639286 0.85598710 1.24787000 6.22515806 18.06787667 14.02353548 9.98290323 5.93085333 3.63090000

2012 6.11632903 5.10758621 4.90230645 4.98811667 5.07176452 6.03904000 7.93350968 10.93765161 11.62335667 11.73020000

2013 15.60704516 26.04308929 57.50405161 130.34097330 119.96354520 107.82259670 85.65485806 103.63570650 123.40197670 150.72433230

2014 857.18219680 662.16510710 591.21047100 460.22734000 484.83181940 612.33593330 615.57314840 535.34762260 442.93792000 361.53317740

2015 248.29459350 233.90223930 274.38047780

Nov Dec

2010 0.25932667 0.24380645

2011 2.68177333 3.48540968

2012 11.48715667 13.36046452

2013 535.00092670 801.73663230

2014 365.00915000 340.49666450

2015

# 3. Check the object tprice and see if you have the correct prices and corresponding months. Plot the object tprice.

#At which year does the price of Bitcoin have a peak? What is the start and end month/year of tprice? What is the frequency of tprice?

**Code**:

tprice

plot(tprice)

start(tprice)

end(tprice)

frequency(tprice)

**Ans**:

#2014

#2010 7 / 2015 3

# 12

# 4. Create a subset of object tprice. Set the start month as July-2013 and end month as Feb-2015. Plot this subset.

#Hint: Use window method

**Code**:

x <- window(tprice,start=c(2013,7), end=c(2015,2))

plot(x)

# 5. Try smoothing and plotting and data using function ma( ). Try k=5, 10 and 15 respectively,

#do you see a smoother plotted curve with increasing k value?

**Code**:

library(forecast)

k<- 5

fit <- ma(ts,k)

head(fit)

mean(fit, na.rm=TRUE)

plot(fit)

k<- 10

fit <- ma(ts,k)

head(fit)

mean(fit, na.rm=TRUE)

plot(fit)

k<- 15

fit <- ma(ts,k)

head(fit)

mean(fit, na.rm=TRUE)

plot(fit)

**Ans**:

# yes the curve is better

# 6. Next try seasonal decomposition using stl().What trend do you see in the data?

**Code:**

lts <- log(tprice)

fit<-stl(lts, s.window="period")

head(fit$time.series)

head(exp(fit$time.series))

**Ans**:

seasonal trend remainder

Jul 2010 0.19761298 -3.142652 0.21605712

Aug 2010 0.02725406 -2.723143 -0.04445147

Sep 2010 -0.25066476 -2.303633 -0.22820630

Oct 2010 -0.37972850 -1.884615 0.04586274

Nov 2010 -0.16042340 -1.465597 0.27635412

Dec 2010 -0.14184077 -1.049512 -0.22002794

seasonal trend remainder

Jul 2010 1.2184907 0.04316816 1.2411733

Aug 2010 1.0276288 0.06566805 0.9565220

Sep 2010 0.7782832 0.09989523 0.7959600

Oct 2010 0.6840471 0.15188746 1.0469307

Nov 2010 0.8517831 0.23093997 1.3183146

Dec 2010 0.8677594 0.35010860 0.8024964

# Its constant till 2013, it rises in 2014 and declines in 2014

# 7. Visualize the seasonal decomposition by using seasonplot functions on the object tprice.

#What are the price trends of Bitcoin in year 2013 and 2014? Do these two years share the same trend?

**Code:**

seasonplot(tprice, year.labels = TRUE)

**Ans:**

# bitcoin increases exponentially in 2013 and declines in 2014.

# The trends aren't the same

# 8. Build a simple exponential smoothing forecasting model using time series object tprice with model = "ANN".

#What is the value of alpha parameter, ie smoothing parameter for the level?

**Code**:

ts<- log(tprice)

fit <- ets(ts, "ANN") #Simple or single exponential forecasting

fit

accuracy(fit)

**Ans**:

# alpha = 0.9999

# 9. Use the forecast() function to predict the time series one step into the future.

#What is the average predicted Bitcoin price for April-2015? What is the 95% confidence interval for this prediction value?

#Plot this prediction with "Month" as x label and "Price" as y label.

**Code:**

predicted <- forecast(fit, 5)

predicted

plot(predicted, xlab = "Month",ylab = "Price")

**Ans**:

# 5.6145

# 6.505540

# 10. Check the accuracy of this simple model for time series forecasts.

#What do RMSE, MAE and MAPE stand for? What value of mean absolute percentage error does this model generate?

**Code**:

accuracy(predicted)

**Ans**:

#6.505540

# RMSE - root mean square error, MAE - mean absolute error , mean absolute percentage error

#36.1928

# 11. Log transform the data and save it as ltprice. B

#Build an exponential smoothing model with Level, Slope, and Seasonal components with ltprice and model="ZZZ".

#Check the model. What are the values of smoothing parameters for the level, trend, and seasonal components?

**Code**:

ltprice <- log(tprice)

fit <- ets(ltprice, model="ZZZ")

fit

**Ans**:

#alpha = 0.9997

#No beta

#No gamma

# 12. Use forecast() function to forecast the Bitcoin price for the next 5 months.

#Plot the prediction with "Month" as x label and "Price" as y label.

#Transform the mean, lower and upper prices of the prediction using exponential function to the actual predictions.

#What is the average predicted Bitcoin price for April-2015? What is the 95% confidence interval of the price?

**Code**:

Pred <- forecast(fit,5)

Pred

plot(Pred,main ="forecast for next 5 months", ylab = "Price", xlab = "Month")

Pred$mean

Pred$upper

Pred$lower

**Ans**:

#5.6145

# 668.8368 - 112.55698

# 13. Check the accuracy of this model for time series forecasts. What value is this model's mean absolute percentage error?

**Code**:

accuracy(Pred)

**Ans**:

#31.29668

# 14. Import library tseries. Decide the best d value for object ltprice using ndiffs( ).

#What is the best d value for our time series object? Then do the differencing of the time series object using diff( ).

#Plot the time series object after differencing. Does it look like there is trend in time series after differencing?

**Code**:

library(tseries)

ndiffs(ltprice)

dNile <- diff(ltprice)

ndiffs(dNile)

plot(dNile)

**Ans**:

#0

#do not see a trend

# 15. Evaluate the assumption of stationarity using Augmented Dickey-Fuller (ADF) test.

#Do we have a stationarity time series object based on the test results?

**Code**:

adf.test(dNile)

**Ans**:

#yes

# 16. Fit an ARIMA model with p = 2, d=1 and q=1. What is the AIC value of this model? Then check the accuracy of the model.

#What is the value of MAPE?

**Code**:

fit<- arima(ltprice, order = c(2,1,1))

fit

accuracy(fit)

**Ans**:

#AIC = 55.63

# 30.55432

# 17. Evaluate the model fitness by check the residuals using qqnorm and qqline functions.

#Does the residuals fall along the line? What can we learn if the residuals fall along the line?

#Use box.test() function to check whether autocorrelations are all zero. What can you interpret from the box.test() results?

**Code**:

qqnorm(fit$residuals)

qqline(fit$residuals)

Box.test(fit$residuals, type="Ljung-Box")

**Ans**:

#yes

# the residuals should be normally distributed with mean 0

# no correlation and normally distributed

# 18. Forecast three months Bitcoin prices with this ARIMA model. What is the predicted average Bitcoin price for April-2015?

**Code:**

predicted <- forecast(fit, 3)

predicted

**Ans**:

# 0.2430621

# 19. Use an automated ARIMA forecasting model for the object ltprice.

#hat are the values for p, d and q? Compare this model and the one from Q16 based on AIC.

#Which of the two models is better base on AIC values?

**Code**:

fit<-auto.arima(ltprice)

fit

**Ans**:

#p =(0,1,1) d = (0,0,1) q=[12]

# AIC = 51.62, auto AIC is better than normal AIC (AIC = 55.63)