simple forecasting methods

Keshav Pokhrel

Jan 25, 2015

Average method

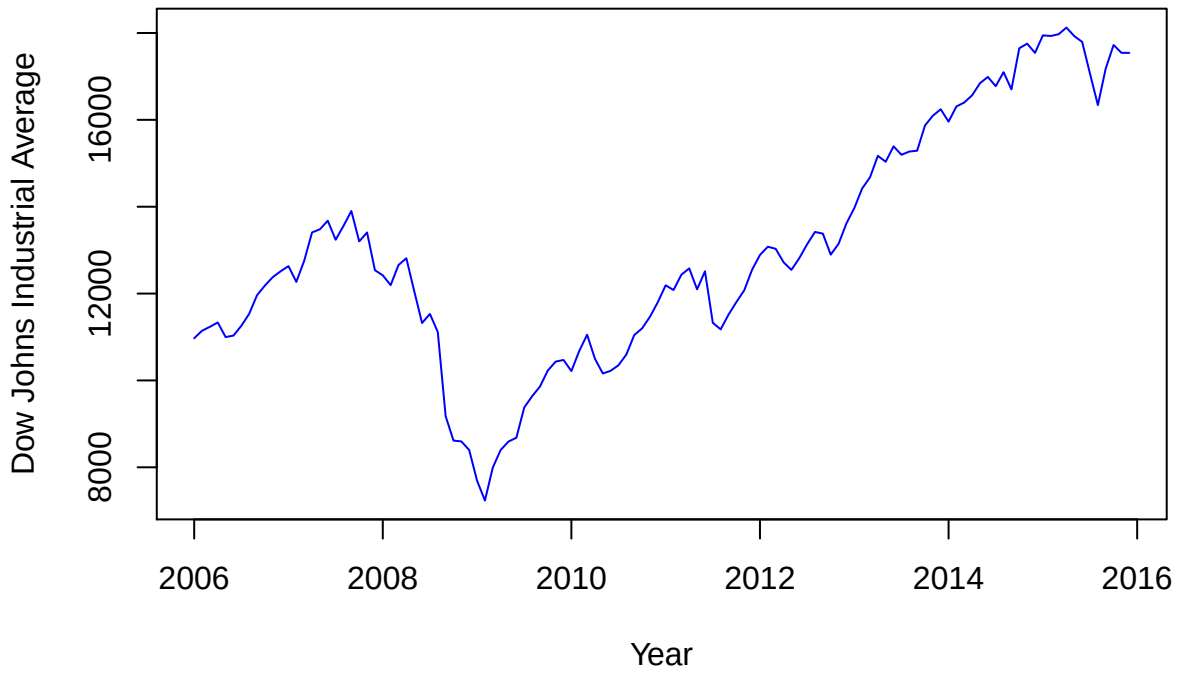
We are going to discuss about Dow Johns Industrial Average data from jan 2006 to Dec 2015. Let

```
setwd("~/Documents/Winter2016/time_series/ts_data_collection")
# this is a location of my data file.
#You can go to session- setup the working directory and copy the same directory from the R-console.
#OR directly type a directory. It is basically a location of data folder
# Make sure to write  the directory within Quotation
# You need to setup a correct directory to run this document

dj<-read.csv("DJIA_monthly.csv", header=FALSE) # there is no column name in the data
```

```
# commenet =NA will remove all the hastags from R output

dj<-dj[,2]
dj1<- ts(dj, start=c(2006,1), end=c(2015,12), frequency=12)
is.ts((dj1))
```

```
[1] TRUE
```

```
plot(dj1, col="blue",  xlab="Year", ylab="Dow Johns Industrial Average")
```



Average Method: Frecaast of all **future** values is the average of all th observed values.

```
The downloaded binary packages are in
    /var/folders/vp/qtt33lh91qvctncrljyk7lg42zbmbl/T//RtmpRylLZV/downloaded_packages
```

```
dj2<- window(dj1, start=c(2006,1), end=c(2012,12), frequency=12)
meanf(dj2, h=5) #h is a forecast horizon
```

```
         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Jan 2013        11504.6 9449.498 13559.71 8340.481 14668.73
Feb 2013        11504.6 9449.498 13559.71 8340.481 14668.73
Mar 2013        11504.6 9449.498 13559.71 8340.481 14668.73
Apr 2013        11504.6 9449.498 13559.71 8340.481 14668.73
May 2013        11504.6 9449.498 13559.71 8340.481 14668.73
```

Naive Method

All forecast are set to have the value of last observation. This method is only appropriate for time series data.

```
naive(dj2, h=5)
```

```
         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Jan 2013       13615.32 13049.72 14180.92 12750.30 14480.34
Feb 2013       13615.32 12815.44 14415.20 12392.00 14838.64
Mar 2013       13615.32 12635.67 14594.97 12117.07 15113.57
Apr 2013       13615.32 12484.11 14746.53 11885.29 15345.35
May 2013       13615.32 12350.59 14880.05 11681.09 15549.55
```

Seasonal Naive Method

Useful method for highly seasonal data. In this case, we set each forecast to be equal to the last observed value from the same season of the year.

```
snaive(dj2, h=5)
```

```
         Point Forecast     Lo 80    Hi 80    Lo 95    Hi 95
Jan 2013       12889.05 10070.859 15707.24 8578.998 17199.10
Feb 2013       13079.47 10261.279 15897.66 8769.418 17389.52
Mar 2013       13030.75 10212.559 15848.94 8720.698 17340.80
Apr 2013       12721.08  9902.889 15539.27 8411.028 17031.13
May 2013       12544.90  9726.709 15363.09 8234.848 16854.95
```

Drift method

Amunt of change over time is called drift. Let T be given time series period. Forecast for time $T+h$ is given by:

$$y_T + \frac{h}{T-1}\sum_{t=2}^{T}(y_t - y_{t-1}) = y_T + \frac{h}{T-1}(y_2 - y_1 + y_3 - y_2 + ... + y_{T-1} + y_{T-2} + y_T - y_{T-1})$$

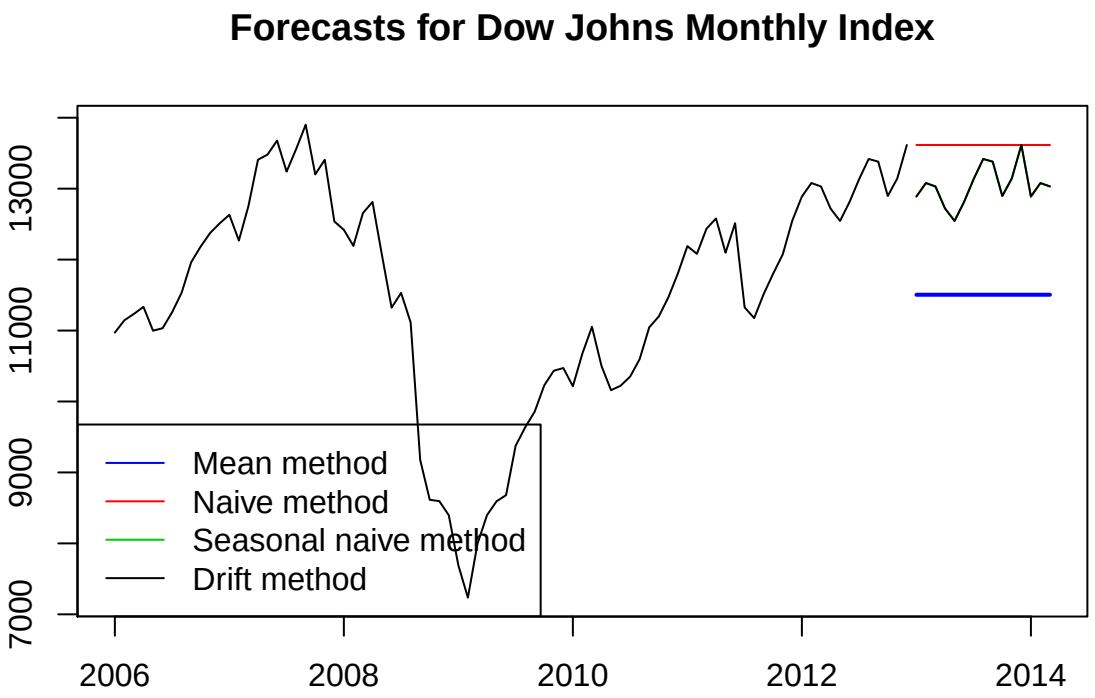
$$y_T + \frac{h}{T-1}\sum_{t=2}^{T}(y_t - y_{t-1}) = y_T + h\left(\frac{y_T - y_1}{T-1}\right)$$

```
rwf(dj2, h=5, drift=TRUE)
```

```
         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Jan 2013       13647.18 13076.21 14218.14 12773.96 14520.39
Feb 2013       13679.03 12866.77 14491.29 12436.79 14921.28
Mar 2013       13710.89 12710.24 14711.54 12180.53 15241.25
Apr 2013       13742.75 12580.60 14904.89 11965.40 15520.10
May 2013       13774.60 12467.84 15081.37 11776.08 15773.13
```

Summary

```
dj2<- window(dj1, start=c(2006,1), end=c(2012,12), frequency=12)
djfit1 <- meanf(dj2, h=15) #h is a forecast horizon
djfit2 <- naive(dj2, h=15)
djfit3 <- snaive(dj2, h=15)
djfit4 <- snaive(dj2, h=15)
plot(djfit1, plot.conf=FALSE,
  main="Forecasts for Dow Johns Monthly Index")
lines(djfit2$mean,col=2)
lines(djfit3$mean,col=3)
lines(djfit4$mean,col=1)
legend("bottomleft",lty=1,col=c(4,2,3,1),
  legend=c("Mean method","Naive method","Seasonal naive method", "Drift method"))
```



Box-Cox Tranformation

Mathematical Adjutment

Calender adjustment

Population adjustment

Inflation adjustment

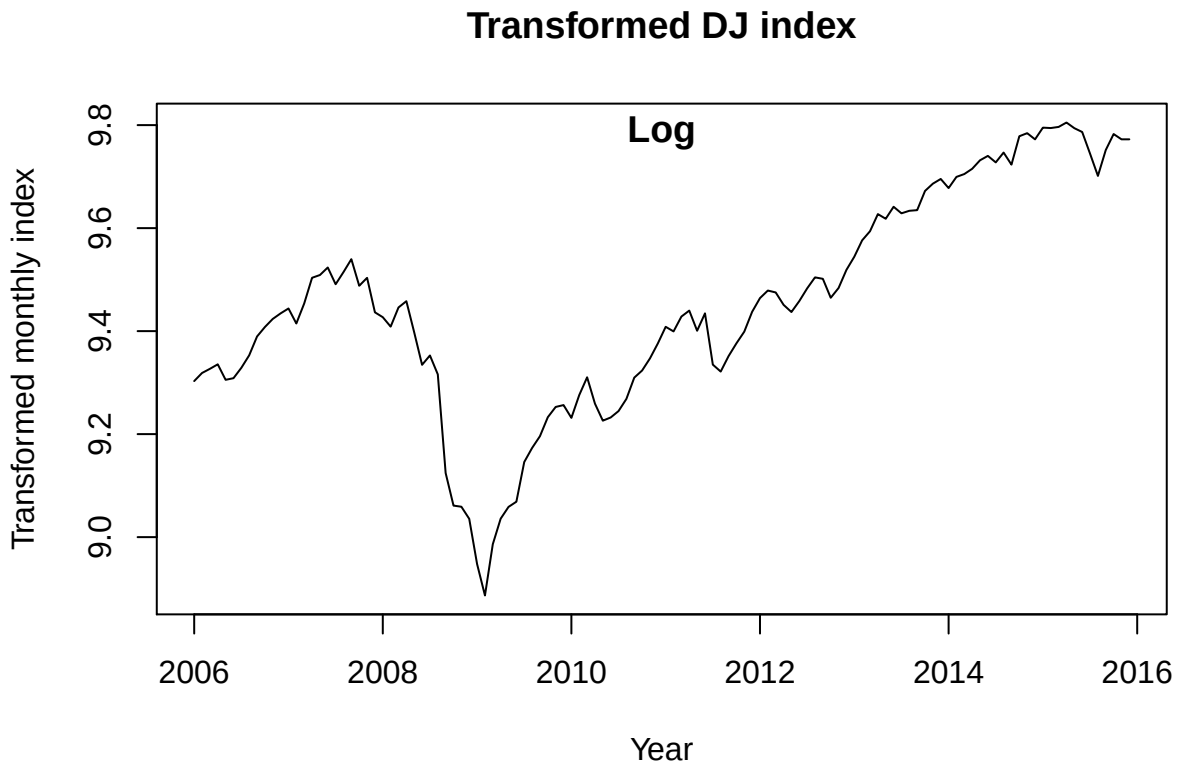
Mathematical Transformation

Useful mathematical transformation are logarithmic and power transformtaion

The Box-Cox power transformation is defined as:

$$w_t = \begin{cases} log(y_t) & if \quad \lambda = 0 \\ \frac{({y_t}^{\lambda} - 1)}{\lambda} & otherwise. \end{cases} \tag{1}$$

```
plot(log(dj1), ylab="Transformed monthly index",
 xlab="Year", main="Transformed DJ index")
title(main="Log",line=-1)
```



After transferring the data we need to reverse the transformation (or back-transform) to obtain forecasts on the original scale. The reverse Box-Cox transformation is given by

$$y_t = \begin{cases} exp(w_t) & if \quad \lambda = 0 \\ (\lambda w_t + 1)^{\frac{1}{\lambda}} & \text{otherwise.} \end{cases} \tag{2}$$

```
# The BoxCox.lambda() function will choose a value of lambda for you.
lambda <- BoxCox.lambda(dj1) # = 1.99 ~  # it looks like square function works well
lambda
```

```
## [1] 1.999924
```

```
plot(BoxCox(dj1,lambda))
```