Understanding and Predicting Stock Prices Behavior using Markov Chain

Authors:

- Srijit Mukherjee (developed the algorithm, and wrote the code)
- Pranjal Verma (applied the code on market data, and reported the results),

Theory

Introduction

Here, the stock market movement is modelled as a discrete state and discrete time

markov chain process. I take daily data to get an estimate of the unknown paramaters of the transition probability matrix. Then, based on the past data, I try to understand whether the behaviour of the stock is behaving as expected by taking the recent data. I also explore the meaning of the parameters based on the data of the best and the worst stock market movements.

Mathematical Formulation

I model a single stock price as a real valued continuous random variable X_t which is changing with time t.

I will convert (X_t, X_{t-1}) into a discrete random variable in the following manner.

Define the following **equation 1**

$$Y_t = \begin{cases} 1(up) & X_t \geq 1.005 X_{t-1} \\ -1(down) & X_t \leq 1.005 X_{t-1} \\ 0(same) & otherwise \end{cases}$$

A real valued discrete random variable Y_t is formed which is changing with time t.

I will consider the random variables Y_t at discrete time points $t_0 = 0, t_1 = 1, t_2 = 2, \cdots$

Assumptions

I have assumed the following conditions on the random variables $Y_0, Y_1, Y_2, Y_3, \cdots$

- $P\left(Y_{n+1} = s \mid Y_0 = y_0, Y_1 = y_1, \ldots, Y_n = y_n\right) = P\left(Y_{n+1} = s \mid Y_n = y_n\right)$

The first condition means, that the stock price is dependent on the just recent past only. This is known as Markov Property.

- $P\left(Y_{n+1} = j \mid Y_n = i\right) = P\left(Y_1 = j \mid Y_0 = i\right) = p_{ij}$

The second condition means that over the change of the stock market from one state to another on an average remains same all over the time. This is known as time homogenous property.

Transition Probability

p_{ij}, as defined above is called the Transition Probability from $i \to j$. Considering all the probabilities, I can represent them in a matrix called Transition Matrix.

$$\begin{array}{c|ccc} & 1 & 0 & -1 \\ 1 & p_{1,1} & p_{1,0} & p_{1,-1} \\ 0 & p_{0,1} & p_{0,0} & p_{0,-1} \\ -1 & p_{-1,1} & p_{-1,0} & p_{-1,-1} \end{array}$$

Data

Now, I have taken data of past n days on stock market prices, $X_0 = x_0, X_1 = x_1, X_2 = x_2, X_3 = x_3, \cdots, X_n = x_n$.

From the definition here, I derive the values $Y_0 = y_0, Y_1 = y_1, Y_2 = y_2, Y_3 = y_3, \cdots, Y_{n-1} = y_{n-1}$.

Estimation of the transition probabilities

Among these Y_i's, I count the following:

n_{ij} = number of k's for which Y_k goes from i to j in Y_{k+1}.

An estimate of p_{ij}, $\hat{p_{ij}}$ is given as $\frac{n_{ij}}{\sum_i n_{ij}}$, i.e. what proportion of times, $i \to j$ as the stock got out of state i.

Steady State Probabilities

Finally, the steady state probabilities of the stock market movement in the state space is calculated. Based on this, the long term behaviour of the stock market is understood. The steady state probabilities give an idea of how much a stock market movement is gonna stay in which state.

A stock market with more "Up" probability is expected to behave well than a stock market with more "Down" probability.

A stock is predicted to be in a state based on which probability is the largest.

Let's explore that by certain examples.

Exploratory Data Analysis 1

I analyze one year performance of certain stocks, and estimate the steady state probabilities. I try to understand the behaviour of the top 10 best and worst performing stocks, and whether it is matching with our intuition or not.

Top 10 Best Performing Stocks

The top 10 best performing stocks are taken according to the Economic Times.

Their steady state probabilities are given below. Let's try to understand.

	1(up)	0(same)	−1(down)
Brightcom Groups	0.5368852	0.1065574	0.3565574
Tata Tele	0.5240686	0.09793022	0.3780012
RattanIndia Infra	0.545082	0.07786885	0.3770492
Adani Gas	0.4596381	0.1959862	0.3443757
Saregama India	0.4658994	0.2010683	0.3330323
Adani Transmission	0.5217554	0.1305535	0.347691
JSW Energy	0.4714983	0.2003513	0.3281505
Borosil Renewables	0.4180799	0.1271866	0.4547335
CG Power and Industrial Ltd.	0.507933	0.1267109	0.3653561
Trident Ltd.	0.4596253	0.158112	0.3822627
Average	0.485	0.137	0.361

Bottom 10 Worst Performing Stocks

The bottom 10 worst performing stocks are taken according to the Economic Times.

Their steady state probabilities are given below. Let's try to understand.

	1(up)	0(same)	−1(down)
AstraZeneca	0.2213115	0.3893443	0.3893443
Alembic Pharma	0.3319516	0.307185	0.3608635
Granules India	0.3968601	0.2049129	0.398227
Biocon	0.3297225	0.3221336	0.348144
CreditAccess Grameen	0.3278689	0.2540984	0.4180328
Aurobindo Pharma	0.331836	0.3007704	0.3673936
AmaraRaja Battries	0.3155738	0.3606557	0.3237705
Bayer Cropsc.	0.3288216	0.3189919	0.3521865
Bandhan Bank	0.3836735	0.2081633	0.4081633
Jubilant Life	0.3373984	0.2520325	0.4105691
Average	0.325	0.287	0.364

Observations

I get a really good observation. In both the worst performing stock markets and good performing stock markets, the down state has almost same stable state probabilities. The

thing that is really making the difference is the distribution of the probabilities in the up state and the same state.

Good Stocks tend to be in the "up" zone more, relative to the same zone.

Bad Stocks tend to remain "neutral" zone, where they are equally distributed.

Predictive Data Analysis 2

I have taken 30 random Indian companies's stock prices, and used 227 days as data to get an idea of the steady state behaviour, and predicted the outcome, based on the maximum probability in the steady state space.

I try to predict and see the prediction correctness based on the Day 228, Average of Day 228 and 229, and Average of the next 7 days.

Behaviour of 30 Indian Companies' Stock Prices

	Predicted	Day 228	Avg Next 2 Days	Avg 7 days
ACC	Up	Up	Up	Up
Adani Enterprise	Up	Up	Up	Up
Dabur	Same	Same	Same	Down
Asian Paints	Up	Down	Down	Down
Bajaj Finance	Up	Up	Up	Up
Bajaj Finserv	Up	Up	Up	Up
Britannia	Same	Down	Down	Up
Dr Reddy	Up	Down	Down	Down
GMR Infra	Down	Down	Down	Base
Grasim	Up	Up	Up	Up
HAL	Up	Up	Same	Down
HCL	Up	Up	Up	Up
HDFC Bank	Up	Down	Same	Down
Hindustan Unilever	Same	Up	Same	Up
Infosys	Up	Same	Same	Up
ITC	Same	Down	Down	Down
Kotak Mahindra Bank	Up	Same	Up	Up
LNT	Up	Up	Up	Up
Mahindra and Mahindra	Up	Down	Down	Down
Marico	Same	Up	Up	Up
Page industries	Up	Down	Down	Down
Pi industries	Up	Up	Up	Up
Saregama	Up	Down	Up	Up
Pidilite	Same	Same	Down	Down
Reliance	Same	Down	Down	Down
Sun Pharma	Up	Up	Up	Up
TCS	Up	Down	Down	Same
TISCO- Tata Steel	Up	Up	Base	Down
Vedanta	Up	Up	Up	Up
Wipro	Up	Down	Down	Up

Observations

You can observe that this algorithm has predicted correctly 50% of the times for the next day and the average of the next two days based on 9.5 months of data. While it decreased a bit to 46.7% when I observe the average behaviour over the next week.

Conclusion

It is told that if you know an algorithm in which you can predict the movement of stock atleast 51%, you will be a billionaire, because one can invest a lot of money and with 1% increase, the profit margin can be really high. So, if one gets a 50% predictability, then the algorithm is a fair and good algorithm. I get a good idea of the behaviour of the best and worst behaving stocks, based on the stable state probabilities.

Now, given a stock, I can understand which stock I can safely invest in, and which stock not to invest in, so that the stock behaves good in the long run.

	UP	SAME	DOWN
UP	$p_{up,up}$	$p_{up,same}$	$p_{up,down}$
SAME	$p_{same,up}$	$p_{same,same}$	$p_{same,down}$
DOWN	$p_{down,up}$	$p_{down,same}$	$p_{down,down}$

```
stockname = read.csv("COMPANYNAME.csv")
data = as.numeric(unlist(stockname))
difference = diff(data)
d = 100*(difference/data[-length(data)]) #relative difference
# D = abs(d) #percentage change
# sorted = sort(D) #sort
#We take top 0.5% change as same
#hit = sorted[ceiling(0.1*length(D))] #top 10% mark of data
#Three states
# U (+1) - more positive than 0.5% increase
# S (0) - within 0.5% change
# D (-1) - more negative than 0.5% decrease
# U -> D : number of changes/total number of days
c = NULL
for (i in 1:length(d))
{
  if (d[i] > 0.5)
  {
    c[i] = 1
  }
```

```
  if (d[i] < -0.5)
  {
    c[i] = -1
  }

  if (d[i] < 0.5 &d[i] > -0.5)
  {
    c[i] = 0
  }
}
# {U,S,D}
#
# U->U, U->S, U->D
# S->U, S->S, S->D
# D->U, D->S, D->D
#
# UU = 0,1 = y = 4
# US = -1,1 = -x+y = 3
# UD = -2,1 = -2x+y = 2
# SU = 1,0 = x = 1
# SS = 0,0 = 0
# SD = -1,0 = -x = -1
# DU = +2,-1 = 2x-y = -2
# DS = 1,-1 = x-y = -3
# DD = 0,-1 = -y = -4
#
# c
# diff(c)
x = 1
y = 4
a = c[-length(c)]*y
b = diff(c)*x
main = a+b
UU = sum(main == 4)
US = sum(main == 3)
UD = sum(main == 2)
SU = sum(main == 1)
SS = sum(main == 0)
SD = sum(main == -1)
```

```
DU = sum(main == -2)
DS = sum(main == -3)
DD = sum(main == -4)

numbers = c(UU,US,UD,SU,SS,SD,DU,DS,DD)
transition_matrix = c(UU/sum(numbers[1:3]),US/sum(numbers[1:3]),UD/sum
                      SU/sum(numbers[4:6]),SS/sum(numbers[4:6]),SD/sum
                      DU/sum(numbers[7:9]),DS/sum(numbers[7:9]),DD/sum
#state space
State = c("Up", "Same", "Down")
#transition matrix
ZoneTransition = matrix (transition_matrix,
                         nrow=3, byrow=T, dimnames = list(State, State
library(markovchain)
library(diagram)
#Diagram
plotmat(ZoneTransition, pos =c(1,2), lwd=1, box.lwd =1, cex.txt =0.5,
        box.type = "circle", box.prop =0.5, box.col = "light yellow",
        arr.type = "curved", self.cex =.4, self.shifty =-.01, self.shi
#?plotmat
#creating the markov chain
StockMarket <- new("markovchain", states = State, byrow = T, transitio
steadyStates(StockMarket)
v0 = c(⅓,⅓,⅓ )
v1 = v0*StockMarket
v1
v2 = v1*StockMarket
v2
v3 = v0*(Activity^3)
v3
```