

Stock Market Analysis and Prediction using Hidden Markov Models

Behrooz Nobakht
Student number: 0938505
bnobakht@liacs.nl

Carl-Edward Joseph
Dippel
Student number: 0953083
cdippel@liacs.nl

Babak Loni
Student number: 4040260
bloni@student.tudelft.nl

ABSTRACT

Stock market analysis and prediction is one of the interesting areas in which past data could be used to anticipate and predict data and information about future. Technically speaking, this area is of high importance for professionals in the industry of finance and stock exchange as they can lead and direct future trends or manage crises over time. In this assignment, we tried to take advantage of Hidden Markov Models to analyze, model and predict the required data having the past data.

Keywords

Data Mining, Stock Market Analysis, Prediction, Knowledge Discovery, Hidden Markov Models

1. INTRODUCTION

Stock market analysis and prediction is of great significance for many professionals in the fields of finance and stock exchange. In this assignment, we are supposed to predict future stock data using the past data. This report tries to summarize the experiments in this regard.

Section 2 briefly introduces the problem. Sections 3 and 3.3 try to shortly introduce how Hidden Markov Models can be used in such a problem.

Section 4 discusses in detail the overall approach we took in this assignment. Section 4.1 talks about the data retrieval; Section 4.2 provides a brief statistical report on the data; Section 4.4 discusses the details of training the HMM; Sections 4.5 and 4.6 present the algorithms to compute likelihoods and prediction criteria.

Section 5 provides the experiments' results and Section 6 talks about the technology and implementation overview of the experiments. Finally, Section 7 proposes some future works into the problem.

2. PROBLEM DEFINITION

As [1] mentions, through this experiment, we try to take

advantage of Hidden Markov Models (HMM) to address some interesting problems regarding stock market analysis. Specifically, stock market index prediction is done in this assignment. First, a set of past data is loaded and analyzed; then, an HMM is modelled and trained for the problem model. Afterwards, similar past data are distinguished and used to predict future stock market values.

Stock market data that are used in this assignment are the data from **FTSE 100 INDEX** that is an index over the stock data in UK. Basically, each stock market data is a quadruple (*open, low, high, close*) carrying the meaning that each day the stock market starts its activity, it starts with some *opening* after which during the day it reaches its *highest* or drops down to its *lowest* of the days and then it will stop with a *close* value. Such data seems to be very sensitive for stock workers and business shareholders to predict future stock trends.

In this assignment, we try to estimate the future day's *close* values as precise as possible.

3. HIDDEN MARKOV MODELS

Briefly to delve into the concepts of **Hidden Markov Model (HMM)**, as [8, 10, 5, 12] propose, an HMM is a state machine for a system adherent to a Markov process with **unobserved states**. Specifically, regarding the time series analysis applications, if we denote the *hidden* state at time t as $x(t)$ and the *observation* at the same time as $y(t)$ then the following facts are always true in the HMM:

1. $x(t)$ is dependent only on $x(t-1)$.
2. $y(t)$ is dependent only on $x(t)$.

To define an HMM, we need

States Q : An HMM has a number of states N and it is usually desired to denote the states that the model goes through in time as $\{q_1, q_2, \dots, q_T\}$ with T as the time length of the observations.

Observations O : in any HMM, during time till T , there is a sequence of observations as $\{o_1, o_2, \dots, o_T\}$.

Transition Matrix $A_{N \times N}$: Each element a_{ij} denotes the probability of transition from state i to state j .

Observation Emission Matrix $B_{N \times T}$: in which $b_j(O_t)$ denotes the probability of observing O_t in state j .

Prior Probability $\pi_{N \times 1}$: in which π_i denotes the probability of being in state i when at time $t = 1$.

Technically, an HMM denoted as λ is considered as the triple:

$$\lambda = (\pi, A, B) \quad (1)$$

3.1 Continuous HMM

In some applications such as stock market analysis or speech recognition, the observations are not from of a *discrete* space; this would make the discrete observation O_t to some \vec{O}_t in each state meaning that in each state a series of observations could be received.

Specifically, the observation vector in our assignment would be:

$$\vec{O}_t = (\text{opening}, \text{low}, \text{high}, \text{close}) \quad (2)$$

which are the values of the stock index. Usually, for such HMM's, the representation for the probability density function (pdf) that is used is a mixture of higher-dimensional Gaussian distribuion:

$$b_j(\vec{O}_t) = \sum_{m=1}^M c_{jm} \times N(\vec{O}_t, \vec{\mu}_{jm}, \Sigma_{jm}) \quad (3)$$

in which

- M is the number of observations in each state. Here, according to [10] and Equation 2 we have $M = 4$.
- c_{jm} is the *mixture coefficient* for the m -th mixture in state j having the property $\sum_{i=1}^N c_{jm} = 1$ for all $1 \leq j \leq N$.
- $\vec{\mu}_{jm}$ is the mean vector for the m -th mixture component in state j .
- Σ_{jm} is the covariance matrix for the m -th mixture component in state j .
- $N(\vec{O}_t, \vec{\mu}_{jm}, \Sigma_{jm})$ is a multi-dimensional Gaussian (Normal) distribution.

3.2 Left-Right (Bakis) HMM

As [11, 6] define, a variation of HMM is used called Left-Right HMM (Bakis HMM) in which the basic ideas is that there is no transition from higher order states to lower order states. This idea is strongly useful in time series analysis applications such as stock market analysis and speech recognition.

Technically, if denote Δ as the effect range of states to each other, there are certain rules in the transition matrix A of a left-right HMM:

$$a_{ij} = \begin{cases} 0 & \text{if } j < i \\ 0 & \text{if } j > i + \Delta \\ \text{some value} & \text{otherwise} \end{cases} \quad (4)$$

Additionally regarding the fact that in such applications, the process starts always from some *initial* state; thus, we have:

$$\pi_i = \begin{cases} 1 & \text{if } i = 1 \\ 0 & \text{if } i \neq 1 \end{cases} \quad (5)$$

3.3 Interesting HMM Questions

Classically, there are three problems that are usually concerned when using HMM's in solving the problems. Having $\lambda = (\pi, A, B)$, there are three questions:

1. How to compute the probability of the occurrence of a specific sequence of observations, $P(\vec{O}|\lambda)$, in which $\vec{O} = \{O_1, O_2, \dots, O_T\}$
2. How to choose state sequence q_1, q_2, \dots, q_T that explains best the observation of \vec{O} in the model λ .
3. How to tune parameters (π, A, B) to find a model λ that best matches the observations of \vec{O}

In stock market analysis and prediction, we are facing the first and the third problem. Training HMM is done through the problem (3) and prediction is achieved with problem (1).

4. APPROACH

4.1 Stock Data Retrieval

There are a number of sources of stock and financial data on web including Google Finance and Yahoo! Finance. In the specific case of **FTSE Index**, it seems that Google Finance does not provide the required data format as required. As an alternative, Yahoo! Finance was used to load and store the stock data from 1984. In the application developed for this project, an interface is included to use when needed to load the data beforehand using the algorithms for training and prediction.

4.2 Statistical Report

4.3 Data Scaling

As the statistical report shows in Section 4.2, there are times that the data could be skewed unequally and unevenly over time. Thus, an approach before starting to train the HMM would be to scale the data values to a specific range such as $[-1, 1]$ using the maximum and minimum values in the range that the algorithms are taken effective. Scaling is also proposed and implemented in [9].

4.4 HMM Training

In this problem, there is a sequence of data over time with which we need to train an HMM; Problem (3) in Section 3.3. To train an HMM matching a set of the sequence of stock index data as $\vec{O} = (\text{open}, \text{high}, \text{low}, \text{close})$, the following settings and considerations have been taken into account:

1. **States:** in the experiments, we have $N = 4$; intuitively, it is denoting the stages in time that are allowed in different transitions in the HMM training.
2. **Dimensions:** as a mixture of multivariate Gaussian is utilized in this problem, we have $D = 4$ as the observation vector for each stock date is as $(\text{open}, \text{low}, \text{high}, \text{close})$.
3. **Mixtures:** [10] proposes to have $M = 3$.
4. **Left-Right Delta:** experimentally, we have tried $\Delta = 1$ and $\Delta = 3$.
5. **Prior Probability:** Adhering to the left-right hmm, we have $\pi = (1, 0, 0, 0)$.

6. **HMM Training:** According to [10, 11], we need to use Baum-Welch Learning Algorithm to train the HMM. Baum-Welch training algorithms does its job based on a number of *iterations* to tune and update the parameters of the model, namely the transition probability B .

4.5 Likelihood Update

In the path to prediction, first, there is a need to find the most similar day in stock market data for a specific day so that it could be used to predict the following day's close value. To do so, first, we need to compute the likelihood of previous days in the desired range. When having one day's stock data, it is straightforward to compute the likelihood of that specific day from the HMM. This is Problem (2) in Section 3.3 which is computed using **Forward Backward Algorithm** proposed in [11, 2, 12]. Algorithm 1 overallly depicts the method to compute likelihoods.

Algorithm 1 Likelihood Update Algorithm

Require: Trained HMM $\lambda = (\pi, A, B)$

Require: Likelihood Update Start Date *start_date*

Require: Likelihood Update Range *days*

1. *date* \leftarrow *start_date*
 2. **for** $i = 1$ to *days* **do**
 3. $O(\text{open}, \text{high}, \text{low}, \text{close}) \leftarrow \text{load}(\text{date})$
 4. $p \leftarrow P(O|\lambda)$ {Forward Backward Algorithm}
 5. $\text{persist}(\text{date}, \log_{10}(p))$
 6. *date* $\leftarrow \text{tomorrow}(\text{date})$
 7. **end for**
-

4.6 Prediction

When the likelihood probabilities of different days are computed, the last phase would be to predict some day's close value as the target of this experiment. To do so, we introduce a parameter **likelihood tolerance** denoting the similarity neighborhood that we can accept similar days to the previous day. Through using the likelihood tolerance, we fetch a list of similar days to yesterday's stock data and then we try to find the best guess as the one that has the highest likelihood of all. In this experiment, we used the likelihood tolerance value in range [0.001, 0.01] From this point, prediction is straightforward with calculating the difference of the similar day and yesterday's values and then calculating tomorrow's close value. Algorithm 2 shows the overall pseudocode used for prediction. Along with prediction computation, we calculate also the **MAPE** (Mean Average Percentage Error) measure.

5. EXPERIMENTS AND RESULTS

As an application is developed for this assignment, a sample result from the application is presented here. Figure 1 depicts a sample run from the application in which a prediction of **1000** days from **2007/01/01** is done using an HMM with settings in Table 1.

6. IMPLEMENTATION

6.1 JAHMM

Algorithm 2 Prediction Algorithm

Require: Prediction Start Date *start_date*

Require: Prediction Days *days*

Require: Likelihood Tolerance *tolerance*

1. *sum* \leftarrow 0
 2. *date* \leftarrow *start_date*
 3. **for** $i = 1$ to *days* **do**
 4. *yesterday* \leftarrow *yesterday*(*date*)
 5. $O_y \leftarrow \text{load}(\text{yesterday})$
 6. $O_a \leftarrow \text{load}(\text{date})$
 7. $\text{likelihood}_y \leftarrow \text{load_likelihood}(\text{yesterday})$
 8. $\text{similar}_y \leftarrow \text{find}(\text{yesterday}, \text{likelihood}_y, \text{tolerance})$
 9. $\text{most_similar} = \text{find_best_guess}(\text{similar}_y)$
 10. $\text{tomorrow}_{m.s} \leftarrow \text{load}(\text{most_similar.date})$
 11. $\text{predicted_close} = O_y.\text{close} + (\text{tomorrow}_{m.s}.\text{close} - \text{most_similar.close})$
 12. $\text{sum} \leftarrow \text{sum} + \frac{|O_a.\text{close} - \text{predicted_close}|}{O_a.\text{close}}$
 13. *date* $\leftarrow \text{tomorrow}(\text{date})$
 14. **end for**
 15. $\text{MAPE} \leftarrow \frac{\text{sum}}{\text{days}} \times 100$
 16. **return** Prediction Results + MAPE
-

Parameter	Description	Value
N	Number of states in HMM	4
M	Number of Multivariate Gaussians	3
D	The dimension of each distribution	4
Δ	The left-right HMM parameter	3
<i>tolerance</i>	The Likelihood Tolerance	0.01

Table 1: Sample settings for the run

As HMM is used in this implementation, there is a need to have an API for HMM operations. Among different implementations, we chose to use **JAHMM** Java library, [7].

6.1.1 HMM Algorithms

HMM-related algorithms such as Baum-Welch and Forward Backward we are provided with JAHMM package and no further implementation was done in this regard.

6.2 Extensions to JAHMM

6.2.1 Mixture of Multivariate Gaussian Distribution

As JAHMM does not provide an implementation for the mixture of multivariate Gaussian distribution, one was developed to be used in the experiments.

6.2.2 Left-Right HMM

JAHMM does not implement the concept of a left-right HMM for which an implementation is given in our project.

6.3 Implementation

6.3.1 Project Source

This project is available under *Apache License Version 2.0* in [3]. Easy and complete instructions of how to get the source and run the project is available at [4].

6.4 Application Architecture

7. FUTURE WORKS

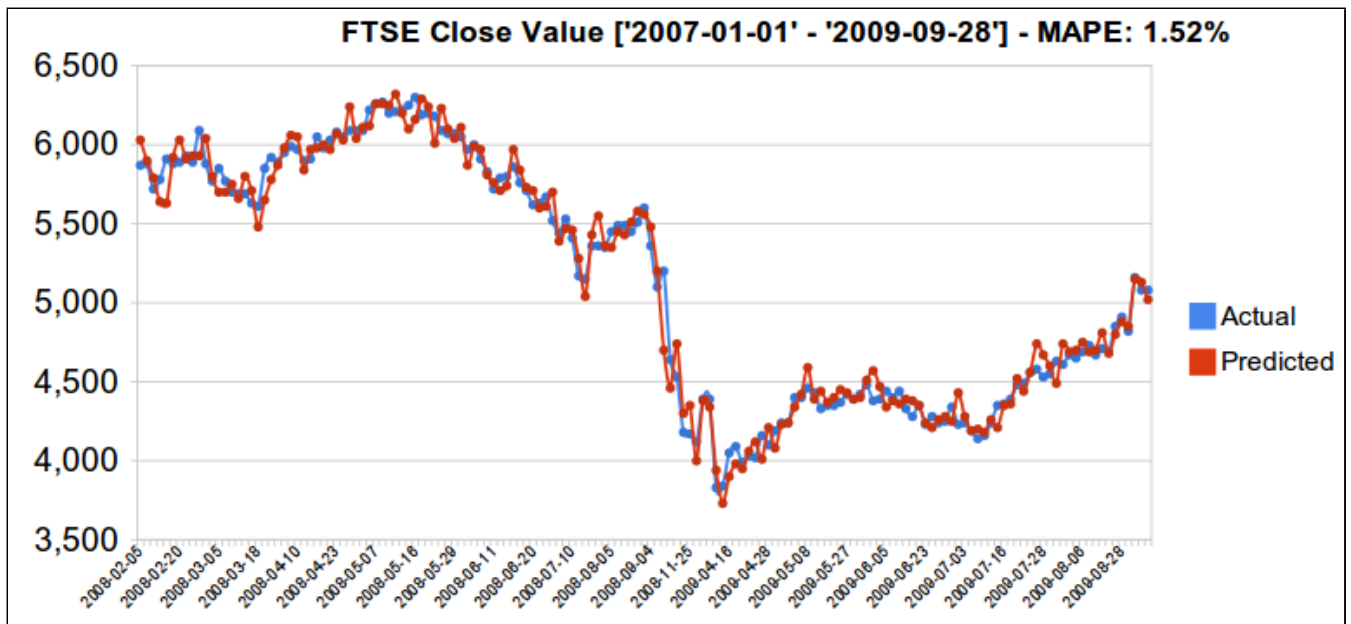


Figure 1: Predicted vs. Actual “close” values from 2007 – 01 – 01 for 1000 days

8. ACKNOWLEDGMENTS

We would like to thank Orr Shomroni and Alexey Gritsenko. Our conversations have been a source for inspiration and were able to contribute (directly and indirectly) to the quality of the report.

9. REFERENCES

- [1] Erwin Bakker. Databases & data mining assignment 4, 2009. <http://www.liacs.nl/~erwin/dbdm2009/assignment04.pdf>.
- [2] Erwin Bakker. Databases & data mining lecture notes, 2009. <http://www.liacs.nl/~erwin/dbdm2009/>.
- [3] Babak Loni Behrooz Nobakht, Carl-Edward Joseph Dippel. Ftse google project, 2009. <http://code.google.com/p/ftse/>.
- [4] Babak Loni Behrooz Nobakht, Carl-Edward Joseph Dippel. Ftse wiki, 2009. <http://code.google.com/p/ftse/w/list>.
- [5] Phil Blunsom. Hidden markov models, 2004.
- [6] Sebastien David, Miguel A. Ferrer, Carlos M. Travieso, and Jesus B. Alonso. gpdshmm: A hidden markov model toolbox in the matlab environment.
- [7] Jean-Marc François. Jahmm, 2009. <http://code.google.com/p/jahmm/>.
- [8] Rafiul Hassan. Phd thesis: Hybrid hmm and soft computing modeling with applications to time series analysis. *Computer Science and Software Engineering Department, University of Melbourne, Australia*, 2007.
- [9] Rafiul Hassan. A combination of hidden markov model and fuzzy model for stock market forecasting. *Elsevier*, 2009.
- [10] Rafiul Hassan and Baikunth Nath. Stockmarket forecasting using hidden markov model: A new approach. *IEEE Computer Society*, 2005.
- [11] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 1989.
- [12] Wikipedia. Hidden markov model, 2009. http://en.wikipedia.org/wiki/Hidden_Markov_model.