Stock Forecasting using Hidden Markov Models

Ayush Jain
Division of Systems Engineering,
Boston University
ayushj94@bu.edu

Prateek Mehta
Department of Electrical and Computer Engineering, Boston University
pmehta59@bu.edu

Ravi Teja Darbha
Department of Electrical and Computer Engineering,
Boston University
rdarbha@bu.edu

Abstract

Stock markets are one of the most complex systems which are almost impossible to model in terms of dynamical equations. The main reason is that there are several uncertain parameters like economic conditions, company's policy change, supply and demand between investors, etc. which drive the stock prices. These parameters are constantly varying which makes stock markets very volatile in nature. Prediction of stock prices is classical problem of non-stationary pattern recognition in Machine Learning. There has been a lot of research in predicting the behavior of stocks based on their historical performance using Artificial Intelligence and Machine Learning techniques like- Artificial Neural Networks, Fuzzy logic and Support Vector Regression. One of the methods which is not as common as the above mentioned for analyzing the stock markets is Hidden Markov Models. Hence, we will be focusing on Hidden Markov Models in this project and compare its performance with Support Vector Regression Model.

Keywords: Hidden Markov Model (HMM), Support Vector Regression (SVR)

1. Introduction

Hidden Markov Models have a strong probabilistic framework for recognizing patterns in stochastic processes. They have been used for analyzing patterns in speech, handwriting and gestures and are still extensively used in those areas. Later HMMs found success in analyzing wide variety of DNA sequences. The underlying idea behind HMMs is that the likelihood of the observations depend on the states of the system which are 'hidden' to the observer. The transition from one state to another is a Markov Process i.e. the next state depends only on the present state, hence the name Hidden Markov Models. States in HMMs are always discrete, while the observations can either be discrete or continuous or both. Stock markets can be viewed as a Hidden Markov Process where the investor can only observe the stock prices and the underlying states which are driving the stock prices are unknown. The observations that we are considering are- daily open, close, high and low for GOOGL stock to build a prototype. In this project, we will consider our observations to be distributed as multivariate Gaussian distribution.

In the past, people have applied HMMs for stock market predictions. Hassan and Nath [1] used fixed state HMMs for predicting some airline stocks by looking for a similar pattern in the past data. Nguyet Nguyen [2] extended the work of Hassan and Nath and used Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) to test the performance of model with respect to number of hidden states. Aditya and Bhuwan [3] used fixed state HMM based MAP estimator to maximize the likelihood of observation of all probable sequences. They compared their results with HMM-Fuzzy logic model and Artificial Neural Network. Another approach is to use Support Vector Regression (SVR) which has been implemented by Cao and Francis [7]. We have implemented an HMM model similar to that implemented by Nguyet Nguyen and compared it's performance with SVR model.

2. Theory

2.1 Hidden Markov Model

HMM is a generative probabilistic model in which the system is considered to be transitioning in certain number of states. The state transition is a Markov Process and hence can be defined by a matrix of state transition probabilities. As previously mentioned, the state sequence is not directly visible but some of the state dynamics can be observed.

2.2 Hidden Markov Models for Financial Time Series analysis

Hidden Markov Models have been a powerful tool for analyzing non-stationary systems. Stock Markets are non-stationary systems and the observations are continuous in nature. Consider O_t be a vector of four elements- daily close, open, high and low and S_t to be the state on day t. The state S_t can be one of the assumed states. The figure 1 shows a typical Hidden Markov Process.

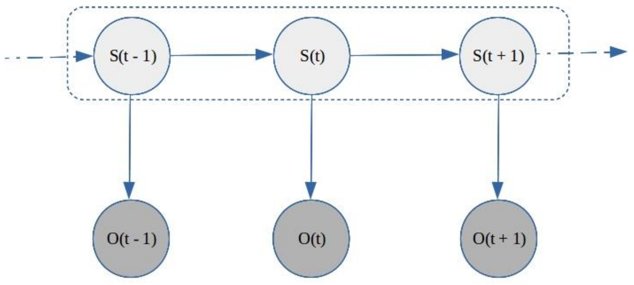

Figure 1

Since the vector O_t takes real values, observations can be modelled as Multivariate Gaussian distributed. The observations are assumed to independent whereas the elements of an observation may be correlated. The state S_t can take only discrete values since HMM is a finite state machine. Let us now define some terminologies that are used to define Hidden Markov Models. We will be using the same notations throughout this project.

Number of observations, T
Latency, K
Number of States, N $(S_t = s_1, s_2, \dots, s_N)$
Observation Sequence, O
Initial State Probability, P_0
State Transition Matrix, $A = [a_{ij}]$ where a_{ij} is the state transition probability from s_i to s_j.
Observation Probabilities, $\mu_i\ \Sigma_i\quad i = 1,2,..N$, where μ_i, Σ_i are the mean and covariance matrix for Gaussian distribution for state i.

The Hidden Markov Model can be represented as

$$\lambda = (A, \mu, \Sigma, P_0)$$

Now the question that needs to be answered is- How to use the Hidden Markov Model? This is answered by solving the following three questions.

- Given the model, how likely it is to observe the given sequence of data?
- Given the model and observations, what is the best hidden state sequence?
- Given the observations, what are the optimal model parameters?

For the first problem, we have used Forward algorithm. The second problem can be solved by Viterbi algorithm. The third problem is solved by Baum-Welch algorithm.

2.3 Prediction of Stock Prices

The main idea for predicting the next day's stock price is to calculate the log-likelihood of K previous observations and comparing it with the log-likelihood of all the previous sub-sequences of same size by shifting the window by one day in the direction of past data. We then identify a day in the past whose log-likelihood of its K previous observations is the closest to the sub-sequence whose next day's price is to be predicted.

$$j = argmin_i \left(|P(O_t, O_{t-1}, O_{t-2}, \dots, O_{t-K} \mid \lambda) - P(O_{t-i}, O_{t-i-1}, O_{t-i-2}, \dots, O_{t-i-K} \mid \lambda)|\right)$$

$$where, i = 1,2, \dots, {}^{T}/_{K}$$

We then calculate the differential price change from the identified day to its next day. This change is then added to the current day's price to get our next day's prediction.

$$O_{t+1} = O_t + (O_{t-j+1} - O_{t-j})$$

Subsequently, after we get the true observation, we include it to our dataset and retune our model parameters in order to ensure that our model doesn't diverge. In short, we fix the size of our sub-sequence and locate another sub-sequence from the past data which exhibits a similar pattern. We then map the behavior of the identified sub-sequence to the sub-sequence being used for prediction.

In order to select a model with optimal number of states, we train a set of models by varying the number of states (N) from the state space G. We have considered the values of number of states in G ranging from $[2, 25]$. We then calculate the negative log-likelihood of the training data used for each of the models and chose the model which has the lowest value. However, this approach tends to prefer a complex model implying that the number of states chosen may tend to a higher value and might result in overfitting. In order avoid this problem, we add a penalty term to the negative log likelihood. Depending on the penalty term chosen, we impose restrictions on the model at varying degree. We have explored two different performance measure metrics, namely, Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC). In AIC, we add the number of model parameters to the negative log-likelihood value to obtain the performance measure of the model whereas in the the case of BIC, we add the product of the number of model parameters and logarithm of the number of observation samples used.

$$AIC = -2\log\left(P\left(O_{train\,|\,\lambda}\right)\right) - 2p$$

$$BIC = -2\log\left(P\left(O_{train\,|\,\lambda}\right)\right) - p.\log(T)$$

$$where, p = N^2 + 2N - 1$$

In our project, we have used BIC as the model's performance measure to select the number of states in our target model.

3. Implementation

The performance metric that we used in this project is Mean Absolute Percentage Error (MAPE) which is defined as

$$MAPE = \frac{1}{N}\sum_{i=1}^{m}\frac{|Predicted(i) - True(i)|}{True(i)}$$

In this project, the main objective was to determine the efficiency of HMMs in predicting the stock prices. We used *hmmlearn*, an open source python library to train the model and calculate the likelihood of the observations. The stocks that we selected are Google Inc., Apple Inc., Qualcomm Inc., and Comcast Corp. We used opening price, closing price, high and low as features for the past 2520 working days (approximately 10 years) when the market was open. We kept aside the recent 100 observations for testing and used rest of the observations for training the model. We predicted the prices for the past 100 days, starting from the 100th day and then using its true observation to retune the model for predicting the 99th day and so on. Therefore, every time we retune the model, the number of training samples will increase by one. First we implemented using fixed model i.e. by fixing the number of states to four. Figures 2-5 shows the stock price predictions for Google using HMM with four states. We calculated the MAPE and plotted the predictions and the actual prices to compare the results. We then optimized our model by selecting the model with lowest BIC value which is a function of number of states.

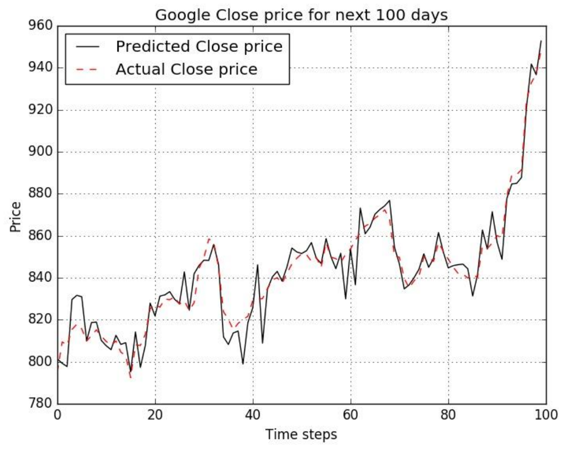

Figure 2

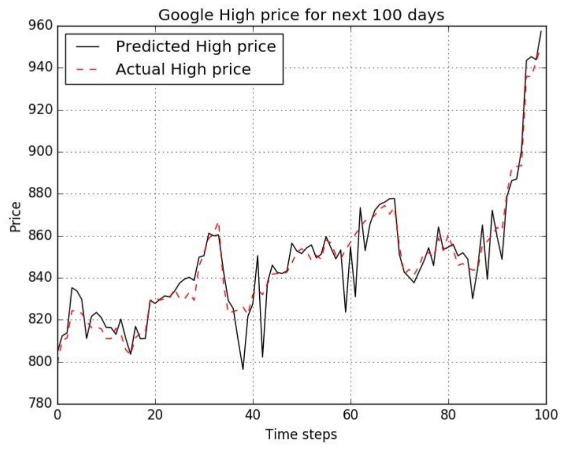

Figure 3

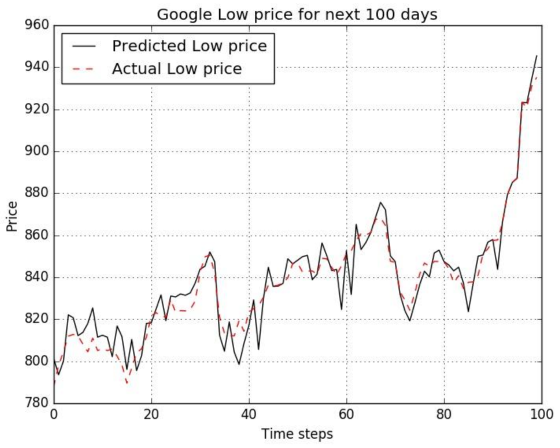

Figure 4

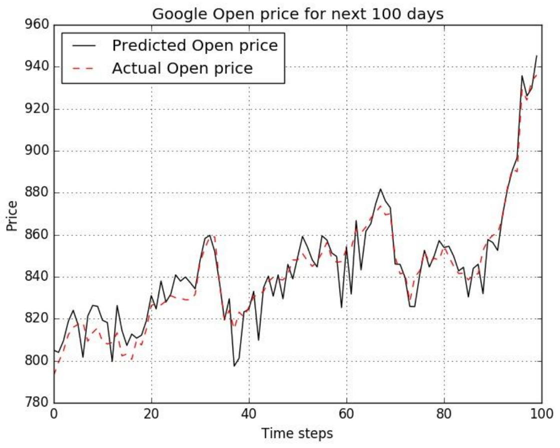

Figure 5

We also used a non-probabilistic model- support vector regression (SVR) to model the stocks. Consider a data set $(x_1, y_1), (x_2, y_2), \dots, (x_T, y_T)$. Unlike Support Vector Machines for classification where y_i is a label, in SVR y_i is a response variable. Using SVR, we tried to find a function which takes in the current day's prices and predicts the next day's close/open/high/ low prices. We formed four models to predict the four prices. Training and testing is exactly same as we did for HMM. We start predictions with 100th day and use its true observation to re-train the SVR to predict for 99th day and so on. To predict the next day's prices, we trained the SVR with the data from the past till last day. We then passed the current day's prices to get the predictions for the next day. We calculated the MAPE and plotted the predictions on the same plots of HMM to compare the results from SVR and HMM. Figures 6-9 shows the true as well as the predicted closing prices for the stocks Apple, Google, Comcast and Qualcomm using HMM as well as SVR models. Tables 1-4 shows the MAPE values for all the four stocks.

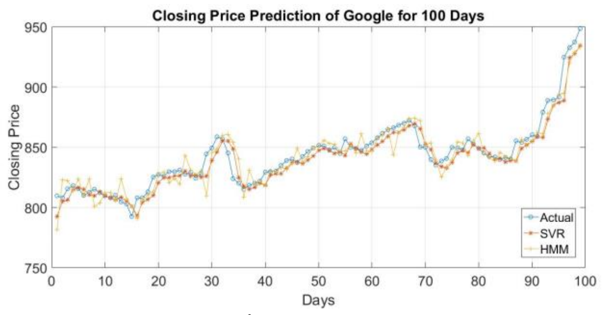

Figure 6

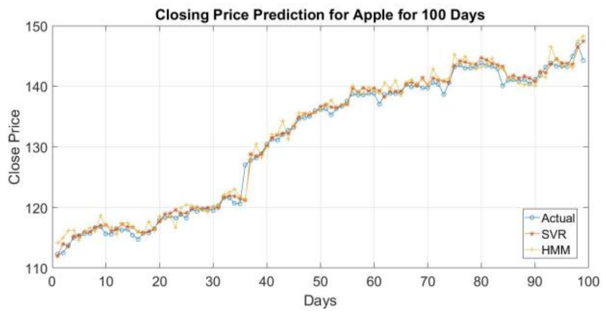

Figure 7

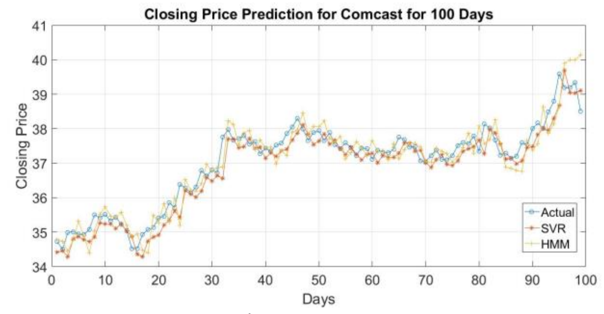

Figure 8

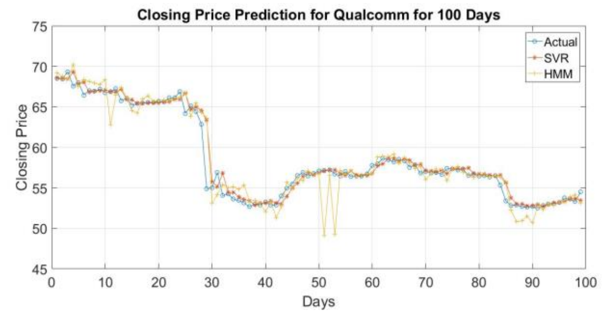

Figure 9

MAPE values for Google

Model	Closing	Open	High	Low
HMM	0.0057	0.0057	0.0061	0.0058
SVR	0.0071	0.0041	0.0053	0.0062

Table 1

MAPE values for Apple

Model	Closing	Open	High	Low
HMM	0.0053	0.0057	0.0044	0.0052
SVR	0.0052	0.0041	0.0053	0.0062

Table 2

MAPE values for Comcast

Model	Closing	Open	High	Low
HMM	0.0059	0.0070	0.0054	0.00586
SVR	0.0075	0.0033	0.0050	0.0047

Table 3

MAPE values for Qualcomm

Model	Closing	Open	High	Low
HMM	0.0124	0.0132	0.0113	0.0122
SVR	0.0098	0.0078	0.0082	0.0106

Table 4

4. Results

We have observed that the predicted values for Open, Close, High, Low closely follow the trends exhibited by its corresponding true values in both the HMM as well as the SVR implementations and the MAPE values were found to be similar. The predictions made using the SVR model was not found to be affected by drastic fluctuations in the stock price. However, the model implemented using HMM was found to be highly sensitive to the fluctuations in stock price. This result is also consistent with the scatter plot of the prediction error on each day for both the models where the spread of the error points in the case of SVR was observed to be concentrated around zero and the and the spread for HMM was observed to be more scatter around zero. Figures 10-13 shows the scatter plots of the error in the prediction of the stock Google using both the HMM and SVR models for each day.

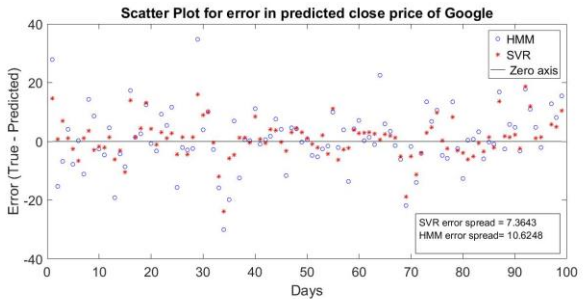

Figure 10

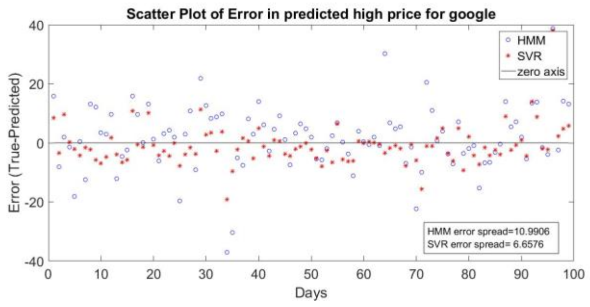

Figure 11

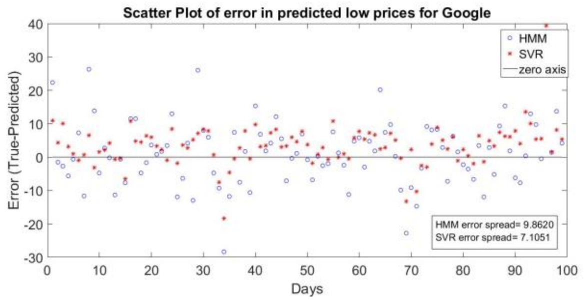

Figure 12

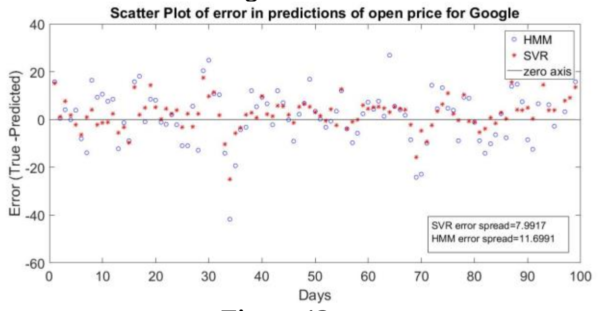

Figure 13

5. Conclusion

Though in general, the observations will be greatly affected by the choice of the model i.e. the number of states in Hidden Markov Models, it did not make significant difference when we tried to find the optimal states using BIC to find best model. Both HMM and SVR give similar accuracy when the next one day prices are predicted. HMM captures the volatility of the stock prices whereas SVR gives more stable predictions. Therefore, HMM can work better for the stocks with high volatility and SVR can work better for stocks which are more stable.

6. Individual Efforts

The individual efforts are as follows

Ravi Teja Darbha

1. Literature survey on Hidden Markov Models for stock price prediction
2. Implementation of HMM for stock prediction using hmmlearn library in Python
3. Detailed explanation of the HMM model for stock prediction and results in the report.

Ayush Jain

1. Literature survey on Hidden Markov Model and SVR for stock price prediction
2. Detailed explanation of Introduction, Theory, Implementation and Conclusion in the report.
3. Explored Kevin Murphy’s HMM toolbox and implemented SVR in MATLAB.

Prateek Mehta

1. Literature survey on SVR for stock price prediction
2. Detailed explanation of Literature survey and generation of various plots in MATLAB in the report.
3. Implementation of SVR in MATLAB

7. References

[1] Hassan, Md. R.; Nath, B.; *Stock Market Forecasting Using Hidden Markov Models: A New approach*. Proceeding of IEEE fifth International Conference on Intelligent Systems Design and Applications 2005.

[2] Nguyet Nguyen, *Stock Price Prediction using Hidden Markov Model,* Youngstown State University, Ohio, USA

[3] Aditya Gupta, Bhuwan Dhingra. *Stock Market Prediction using Hidden Markov Models*. IEEE

[4] Gaurav Kshirsagar, Mohit Chandel, Shantanu Kakade, Rukshad Amaria. *Stock Market Prediction using Artificial Neural Networks.* International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), Volume 5, Issue 5, May 2016

[5] Ypke Heimstra, *A stock Market Forcasting Support System Based on Fuzzy Logic.* 1994 Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences, Volume 3.

[6] Chi-Yuan Yeh, Chi-Wei Huang, Shie-Jue Lee, *A multiple-kernel support vector regression approach for stock market price forecasting.* Expert Systems with Applications, Volume 38, Issue 3, March 2011

[7] L. J. Cao, Francis E. H. Tay: *Support Vector Machine with adaptive parameters in financial time series forecasting*. IEEE transactions on neural networks, vol. 14, no. 6, november 2003