Stock Market Trend Analysis Using Hidden Markov Model and Long Short Term Memory

Junbang Huo*†, Yulin Wu*†, Jinge Wu*†
*Likelihood Lab
†Sun Yat-sen University，Xi'an Jiaotong-Liverpool University

$\{huojb3, wuylin6,\}@mail2.sysu.edu.cn, Jinge.Wu16@student.xjtlu.edu.cn$

***Abstract*---This paper intends to apply the Hidden Markov Model into stock market and to predict the ups and downs in the future. Moreover, four different methods of improvement, which are GMM-HMM, XGB-HMM, GMM-HMM+LSTM and XGB-HMM+LSTM, will be discussed later with the results of experiment respectively. After that we will analyze the pros and cons of different model. And finally, one of the best will be used into stock market to make timing decisions.。**

***Index Terms*---GMM-HMM;XGB-HMM;LSTM**

I. Introduction

HMM and LSTM have been widely used in the field of speech recognition in recent years, which has greatly improved the accuracy of existing speech recognition systems. Based on many similarities between speech recognition and stock forecasting, we propose the idea of applying HMM and LSTM into financial markets, and use machine learning algorithms to predict ups and downs of stock market. We explore the effect of the model through experiments. First, we use the GMM-HMM hybrid model method. Then, improve HMM model by using XGB algorithm (XGB-HMM). Next, we build a long-short term memory model (LSTM), then use GMM-HMM and LSTM together, also compared results with XGB-HMM and LSTM. By comparing the results of the four experiments, the model which is most suitable for the stock market wil be summarized and applied. This article will explain the core of each algorithm and the practical operation, and we will open the source of code for reference and learning on GitHub. [1].

II. Data Process

A. Feature engineering

1) Construction of Y feature: We need to create a feature Y which can reflect the trend of stock market in order to predict the ups and downs of stock price. Thus a method named the triple barrier method[5] can be well used in this situation.The triple barrier method is put forward by Marcos for the first time, and it is an alternative labelling method see in Fig.1.

[1]https://github.com/JINGEWU/Stock-Market-Trend-Analysis-Using-HMM-LSTM

Notation
T : length of the observation sequence
N : number of states in the model
M : number of observation symbols
A : state transition probabilities
B : emission probability matrix
π : initial state distribution
$O = \{O_0, O_1, .., O_{T-1})\}$: observation sequence
$S = \{S_0, S_1, ..., S_TT - 1\}$: state sequence
$\lambda = (A, B, \pi)$: HMM model
$socre\_plot$: score function

Table I
Notation

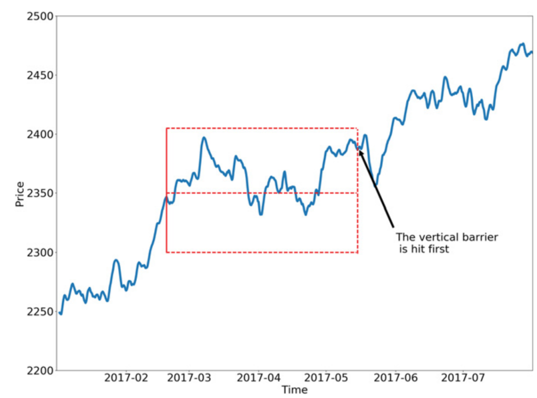


Figure 1. the triple barrier method 结构 [5]

It sets three dividing lines and marks them according to the first barrier touched by the path.

First, we set two horizontal barriers and one vertical barrier. The two horizontal barriers are defined by profit-taking and stop loss limits, which are a dynamic function of estimated volatility (whether realized or implied). The third barrier is defined in terms of number of bars elapsed since the position was taken (an expiration limit). If the upper barrier is touched first, we label the observation as a 1. If the lower barrier is touched first, we label the observation as a -1. If the vertical barrier is touched first, we have two choices: label the observation as 0,[5]which is shown in Fig.2.

The triple barrier method reflects the rise and fall of the stock's future price.。

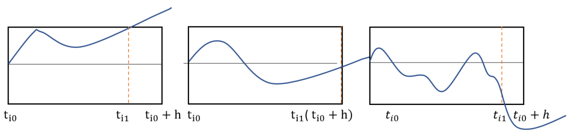


Figure 2. Label t_{i0} as 1,0 and -1

One problem with the triple-barrier method is path-dependent. In order to label an observation, we must take into account the entire path spanning $[t_{i0}, t_{i0}+h]$, where h defines the vertical barrier (the expiration limit).We will denote$t_{i1}$the time of the first barrier touch, and the return associated with the observed feature is t_{i0}, t_{i1}. For the sake of clarity, $t_{i1} \le t_{i0}+h$, and the horizontal barriers are not necessarily symmetric.[5]。

2) Construction of the observation sequence: Our data of observation sequence is divided into 8 different types:

Table	
Type	Meaning
Market Factor	PreClosePrice，OpenPrice ClosePrice，DealAmount etc
Quality factor	50 factors such as accounts payable ratio turnover days, management expenses and total operating income
Income risk Factor	10 factors such as Cumulative income and falling fluctuation
Value Factor	13 factors such as cash flow market value ratio income market value ratio, price-to-book ratio and price-earnings ratio
Mood Factor	49 factors such as hand turnover rate,dynamic trading trading volume
Index Factor	49 factors such as exponential moving average and smoothing similarity moving average
Momentum Factor	56 factors such as stock income and accumulation/distribution line
Rise Factor	17 factors such as net asset growth rate and net profit growth rate

Table II
FACTORS

In order to classify different types of observation sequence, we use a score function。

- Fix one factor F, assume the daily data of F is O, train GMM-HMM model, and generate the optimal state sequence S by using Viterbi algorithm.
- Create a count matrix M,,$M \in R^{N\times 3}$,in which M_{ij}represent the frequency of label j corresponding to state i, $i \in \{0,1,...,N-1\}, j \in \{-1,0,1\}$
- Constructing a counting ratio matrix MR according to M where,

$$MR_{ij} = \frac{M_{ij}}{\sum_j M_{ij}}$$

- The accuracy of state i is

$$Acc_i = max_j\{MR_{ij}\} \tag{1}$$

$$i \in \{0,1,...,N-1\}, i \in \{0,1,...,N-1\}, j \in \{-1,0,1\} \tag{2}$$

- The entropy of state i is

$$H_i = -\sum_j MR_{ij} log(MR_{ij}) \tag{3}$$

$$i \in \{0,1,...,N-1\}, i \in \{0,1,...,N-1\}, j \in \{-1,0,1\} \tag{4}$$

- The weights of state I is

$$w_i = \frac{\sum_j M_{ij}}{\sum_i \sum_j M_{ij}} \tag{5}$$

- The score function is

$$score = score\_plot(Acc, H, w) = \sum_i (Acc_i \times \frac{1}{1+H_i} \times w_i) \tag{6}$$

- The higher the score is, the better the feature is.

The features selected by score function is in the table below.

Table	
Market Factor	$log(Tday'sClosePrice - (T-5)day'sClosePrice)$ $log(\frac{HighestPrice}{LowestPrice})$ $log(Tday'sDealAmount - (T-5)day'sDealAmount)$ $\frac{ClosePrice}{PreClosePrice}$ $\frac{OpenPrice}{PreClosePrice}, \frac{HighestPrice}{PreClosePrice}$ $\frac{LowestPrice}{PreClosePrice}$
Quality Factor	DEGM, EBITToTOR, MLEV , CFO2EV , NetProfitRatio , ROA5
Income risk Factor	DDNBT , TOBT
Value Factor	CTP5, SFY12P
Mood Factor	ACD20,ACD6, STM, VOL5, DAVOL5
Index Factor	MTM, UpRVI, BBI, DownRVI, ASI
Momentum Factor	BBIC , PLRC6 , ChandeSD , PLRC12 , ChandeSU , BearPower
Rise Factor	FSALESG , FEARNG

Table III
FEATURES

III. GMM-HMM

A. Introduction

Gaussian Mixture-Hidden Markov Model (GMM-HMM) has been widely used in automatic speech recognition (ASR) and has make much success.

Motivated by its application, our task is: given a observation sequence, find most likely hidden state sequence, and then find relation between hidden state and stock market trend by using Long-Short Term Memory.

We assume:

Observation sequence $O = O_1 O_2 ... O_T$,O_t is the observation of time t.

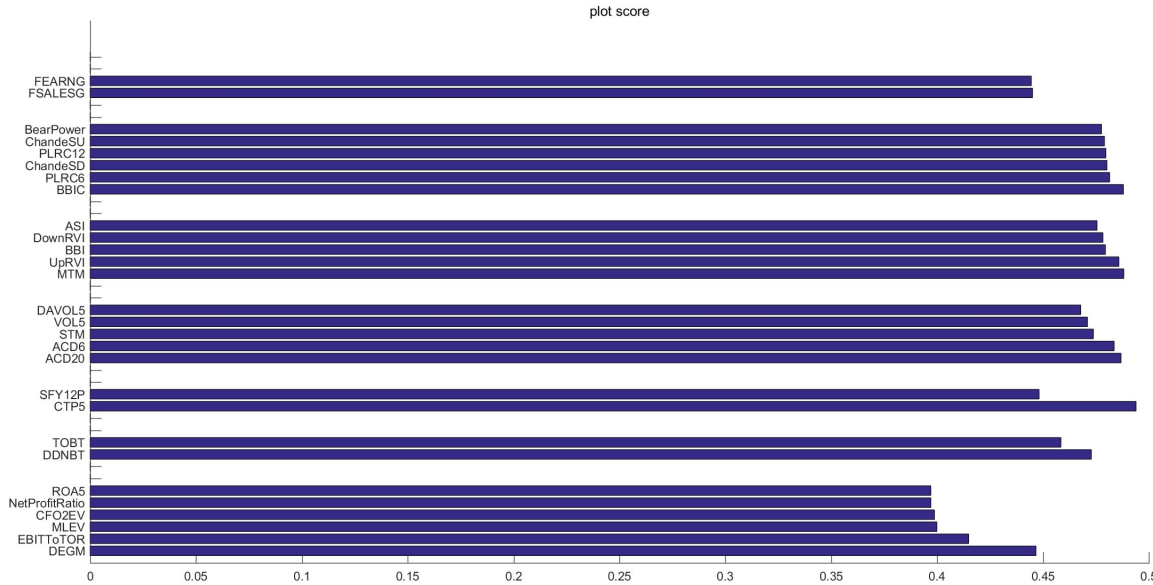


Figure 3. feature socres

Hidden state sequence $S = S_1 S_2 ... S_T$, S_t is the hidden state of time t.

The most likely hidden state is:

$$s = argmax_S P\{S|O\}$$

In this section, we will introduce core algorithms of GMM-HMM in predicting stock price and its practical operation

B. Basic Hypothesis

(1) The state at any time t depends on its state at time t-1, and is independent of any other moment.

$$P\{S_t|S_{t-1}, O_{t-1}, ..., S_1, O_1\} = P\{S_t|S_{t-1}\}$$

(2) The observation of any time t depends on its state at time t, and is independent of any others.

$$P\{O_t|S_T, O_T, ..., S_1, O_1\} = P\{O_t|S_t\} \tag{7}$$

C. GMM-HMM

A GMM-HMM is a statistical model that describes two dependent random processes, an observable process, and a hidden Markov Process. The observation sequence is assumed to be generated by each hidden state according to a Gaussian mixture distribution.[1]

The hidden Markov model is determined by the initial probability distribution vector π, the transition probability distribution matrix A, and the observation probability distribution called emission matrix B. π and A determine the sequence of states, and B determines the sequence of observations.

In GMM-HMM, the parameter B is a density function of observation probability, which can be approximated as a combination of products of mixed Gaussian distributions. By evaluating each Gaussian probability density function weight w_{jk}, mean and covariance matrix, the continuous probability density function can be expressed as:

$$b_j(V_t) = \sum_{k=1}^{M} w_{jk} b_{jk}(V_t), j = 1, ..., N, 0 \leq t \leq M-1$$

In which w_{jk} represents the weight of each part.

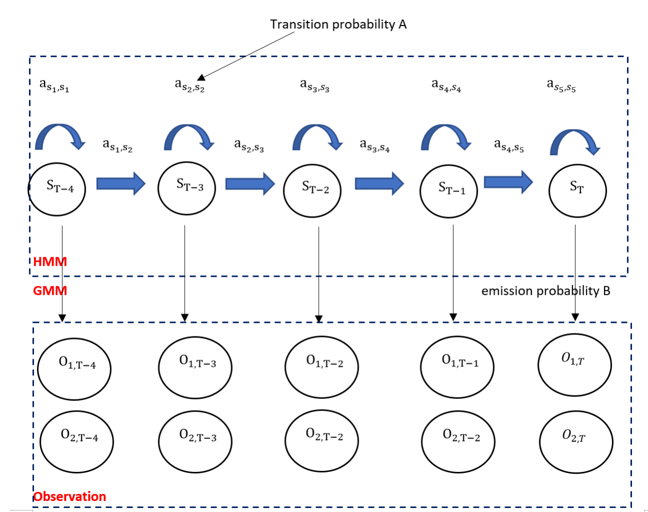


Figure 4. GMM-HMM Structure

D. GMM-HMM algorithm

- Determine the number of hidden states N

- Estimate model parameters $\lambda = (A, B, \pi)$ by using Baum-Welh algorithm at a given observation sequence O. Record model parameters.Record model parameters $\lambda = (A, B, \pi)$ as gmm-hmm.
- Estimate probability $P\{S_t = i\}, i = \{1, 2, .., N\}, t = \{0, 1, .., T-1\}$ by using Viterbi algorithm in gmm-hmm.

E. Results and analysis of experiment

In order to test the effect of our model, we select a number of stocks for model training and determine the various parameters of the model.

For our better understanding, we choose one of the stocks Fengyuan Pharmaceutical (000153.XSHE) which is from 2007-01-04 to 2013-12-17 to do the visualization. The figure is below.

We set N=3, and use market data as a feature, and then predict the state at each moment.

We use market information as feature and set mew model as $gmm - hmm_1$.

Then put the predicted state for time t in $gmm - hmm_1$ and its close price together to make visualizetion.

The results of the training set are shown below.

As can be seen from the figure, red represents the rising state, green represents the falling state, blue represents the oscillation, and intuitively, $gmm - hmm_1$ works well on the training set.

Use the data of the stock Red Sun (000525.XSHE) from 2014-12-01 to 2018-05-21 as a test set, use the predicted state for time t in $gmm - hmm_1$ and its close price together to make visualization.

The results of the test set are shown below.

From the figure 6 we can find that $gmm - hmm_1$ works well on the test set.

Then, we trained seven multi-factor features that were filtered to get the model $gmm - hmm_i, i = 2, 3, ..., 8$, to prepare for the latter LSTM model.

IV. XGB-HMM

A. Introduction

In the above GMM-HMM model, we assume that the emission probability B is a Gaussian mixture model.

Our team believes that in addition to the GMM distribution, we can use the scalable end-to-end tree propulsion system XGBoost to estimate the emission probability B.

In the GMM-HMM model, we assume that the emission probability B is a Gaussian mixture model.

Our team believes that in addition to the Gaussian distribution, we can use the scalable end-to-end tree propulsion system XGBoost to estimate the emission probability B.

In this section, we introduce a XGB-HMM model which exploits XGBoost instead of Gaussian mixture model (GMM) in estimating the emission probabiltiies $P\{O_t|S_t\}$.

In this section, we will explain the training algorithms of XGBoost.

B. Applying XGB into emission probability

1) XGB-HMM: Machine learning and data-driven methods have become very important in many areas. Among the machine learning methods used in practice, gradient tree boosting is one technique that shines in many applications.

The linear combination of trees can fit the training data well even if the relationship between the input and output in the data is complicated. Thus the tree model is a highly functional learning method. Boosting tree refers to the enhancement method which uses the addition model (that is, the linear combination of the basis functions) and the forward stagewise algorithm, and the decision tree as the basis function. It is an efficient and widely used machine learning method.[2]

In this chapter, we introduce a scalable end-to-end tree boosting system called XGBoost, which is used widely by data scientists to achieve state-of-the-art results on many machine learning challenges. Compared with the general GBDT algorithm, XGBoost has the following advantages[6]:

- *Penalize the weight of the leaf node, which is equivalent to adding a regular term to prevent overfitting.*
- *XGBoost's objective function optimization utilizes the second derivative of the loss function with respect to the function to be sought, while GBDT only uses the first-order information.*
- *XGBoost supports column sampling, similar to random forests, sampling attributes when building each tree, training speed is fast, and the effect is good.*
- *Similar to the learning rate, after learning a tree, it will reduce its weight, thereby reducing the role of the tree and improving the learning space.*
- *The algorithm for constructing the tree includes an accurate algorithm and an approximate algorithm. The approximate algorithm performs bucketing on the weighted quantiles of each dimension. The specific algorithm utilizes the second derivative of the loss function with respect to the tree to be solved.*
- *Added support for sparse data. When a certain feature of the data is missing, the data is divided into the default child nodes.*
- *Parallel histogram algorithm. When splitting nodes, the data is stored in columns in the block and has been pre-sorted, so it can be calculated in parallel, which is, traversing the optimal split point for each attribute at the same time.*

2) XGB-HMMtraining algorithm: Referring to the GMM-HMM model training algorithm, our team has obtained the training algorithm of the XGB-HMM model as follows.

Initialize:

- Train one GMM-HMM parameters $\lambda = (A, B, \pi)$, assuming the model is gmm-hmm.
- Calculate $\alpha_t(i), \beta_t(i)$ by using forward-backward algrithm, and then calculate $\gamma_t(i) = P\{S_t = i|O, \lambda\}$ 和 $\gamma_t(i, j) = P\{x_t = q_i, x_{t+1} = q_j|O, \lambda\}$

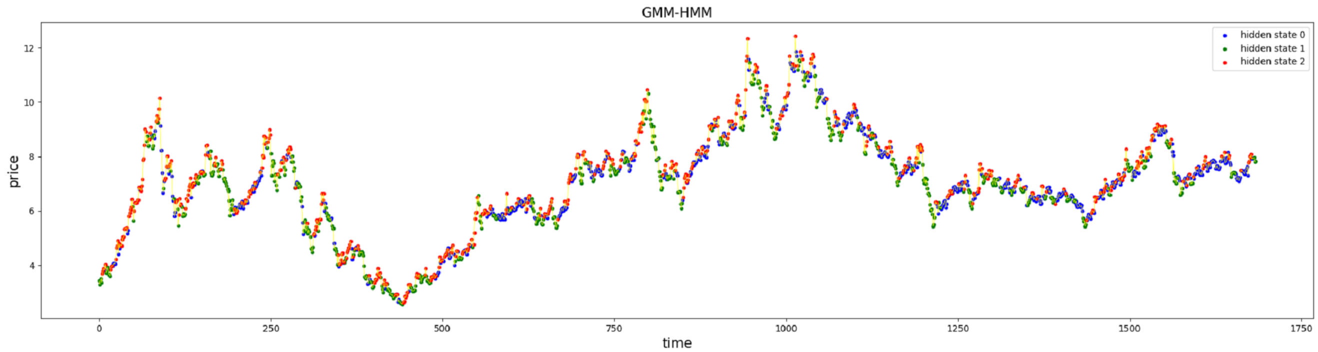


Figure 5. $gmm - hmm_1$training set

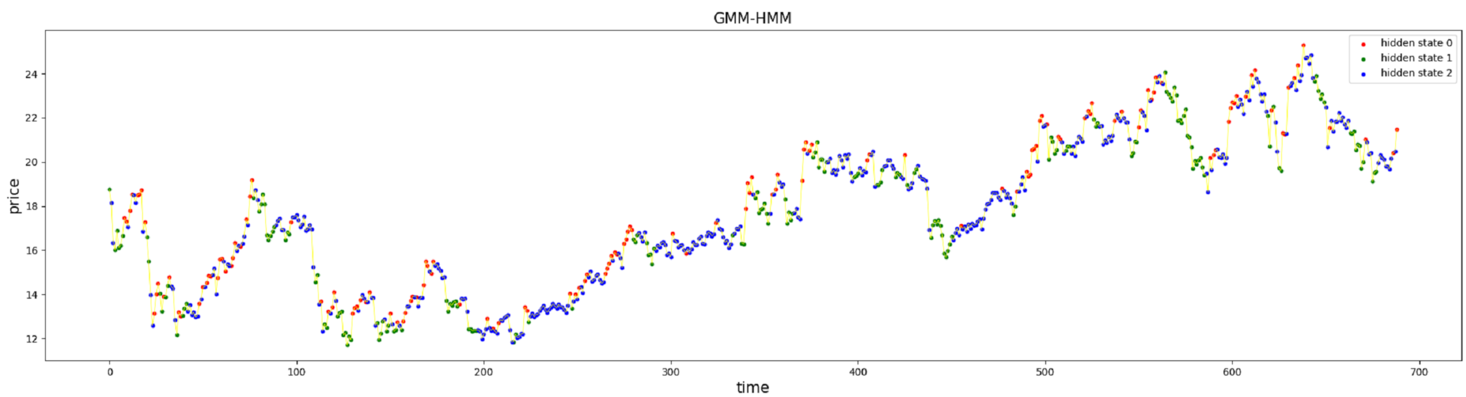


Figure 6. $gmm - hmm_1$test result

for t=1,2,..,T-1 and i = 0,1,..,N-1

$$\alpha_t(i) = P\{O_0, O_1, ..., O_t, x_t = q_i|\lambda\} \tag{8}$$

$$\beta_t(i) = P\{O_{t+1}, O_{t+2}, ..., O_{T-1}|x_t = q_i, \lambda\} \tag{9}$$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P\{O|\lambda\}}$$

$$\gamma_t(i,j) == \frac{a_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P\{O|\lambda\}}$$

Re-estimate：

- According

$$a_{ij} = \frac{\sum_{t=0}^{T-2}\gamma_t(i,j)}{\sum_{t=0}^{T-2}\gamma_t(i)}$$

Re-estimate transition probability A。

- Input O, supervised learning $\gamma_t(i) = P\{S_t = i|O, \lambda\}$ training a XGB model，say XGB_0.
- Use XGB_0 to estimate $\hat{\gamma_t(i)}$, recorded as $\gamma_t(i)_{new}$， and the use

$$b_j(k) = \frac{\sum_{t\in\{0,1,..,T-1\}\&O_t=k}\gamma_t(j)_{new}}{\gamma_t(j)_{new}}$$

to re-estimate emission probability B。

- If $P\{O|\lambda\}$ increases,returen to 2， continuously re-estimate λ；otherwise，finish training.
- Finally, we get $\lambda = (A, B, \pi)$ and XGB_0。

The training algorithm can be summarized as follows.

- Initialize,$\lambda = (A, B, \pi)$.
- Compute $\alpha_t(i), \beta_t(i), \gamma_t(i), \gamma_t(i,j)$.
- Train XGB.
- Re-estimate the model $\lambda = (A, B, \pi)$.
- If $P\{O|\lambda\}$ increases, goto 2.
- Of course, it might be desirable to stop if $P\{O|\lambda\}$ does not increase by at least some predetermined threshold and/or to set a maximum number of iterations.

XGB-HMM training algorithm in Figure 7.

C. XGB-HMM 结果分析

In order to test the effect of our model, we select a number of stocks for model training and determine the various parameters of the model.

For our better understanding, we choose one of the stocks Fengyuan Pharmaceutical (000153.XSHE) which is from 2007-01-04 to 2013-12-17 to do the visualization. The figure is below.

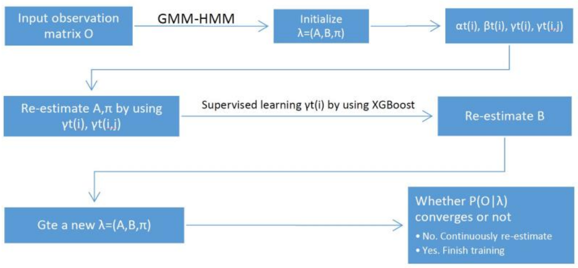


Figure 7. XGB-HMM training algorithm

We set N=3, and use market data as a feature, and then predict the state at each moment and its. The prices should be put together for visual analysis.

The results of the XGB-HMM training set are shown in Fig. 8. The results of the XGB-HMM test set are shown in Fig. 9. The iteration diagram of the XGB-HMM algorithm is shown in Fig. 10.

As can be seen from the Fig. 8, red represents the rising state, green represents the falling state, blue represents the oscillation, and intuitively, XGB-HMM works well on the training set, and the model is $xgb - hmm_0$.

Use the data of the stock Red Sun (000525.XSHE) from 2014-12-01 to 2018-05-21 as a test set, use model as $xgb - hmm_0$ see in Fig.9.

As can be seen from the Fig.10, as the number of iterations increases, log-likelihood increases as well. When iteration reach about 250, log-likelihood tends to be stable. At the same time, as the number of iterations increases, plot score increases.

D. *comparation of GMM-HMM and XGB-HMM*

Fig.11,12 is the comparation of GMM-HMM and XGB-HMM Results on test set and training set.

It can be seen that the result of using the XGB-HMM model is better, which makes the discrimination of the three hidden states higher, which means that XGB may have a better effect in fitting the observed data.

Moreover the results of XGB-HMM tends to be better than GMM-HMM.

E. *strength and weakness of model and its improvement*

Strength:

GMM cannot capture the relationship amoung different observation features, but XGB can.

In the training set, the accuracy of XGB training is 93%. On the test set, the accuracy of xgb training is 87%, and the fitting effect of XGB on the emission probability B is better than that of GMM.

Therefore, the XGB-HMM works better than the GMM-HMM in both the training set and the test set.

Weakness:

In the GMM-HMM model and the XGB-HMM model, we use visualization to roughly observe the relationship between the state sequence S and the Y feature. It is considered that the red state represents the rise, the green state represents the decline, and the blue state represents the oscillation.

However based on the training set of XGB-HMM and the test set results, we can see that the three states and the ups and downs of stock price are not relatively consistent. For example, when the stock price reaches a local maximum and starts to fall, the state is still red, and it is green after a few days.

Our team believes that the previous model did not take into account the probability of the state corresponding to each node. For example, when the state corresponding to a node is state 1, we think that this node must be state 1. But according to the XGB-HMM model, we can get the probability of each node corresponding to each state $P\{S_t = i\}, i = 1, 2, 3$, and then get the probability matrix state_proba.

$$state\_proba = \begin{pmatrix} P\{S_1 = 1\}......P\{S_{T-1} = 1\} \\ P\{S_1 = 2\}......P\{S_{T-1} = 2\} \\ P\{S_1 = 3\}......P\{S_{T-1} = 3\} \end{pmatrix}$$

Next we will use Long-Short Term Memory to find the relation between state_proba and Y.

V. GMM-HMM+LSTM & XGB-HMM+LSTM

A. *Introduction*

The algorithm was first published by Sepp Hochreiter and Jurgen Schmidhuber on Neural Computation.

The Long Short Term Memory model consists of a dedicated memory storage unit that controls the state of each memory storage unit through carefully designed forgetting gates, input gates and output gates. The control of each gate ensures the previous information can continue to propagate backwards without disappearing while the hidden layer is continuously superimposed with input sequence in the new time state.[5]。

LSTM can be regarded as a mapping from X to Y, in which X is a n × k matrix, Y is a vector with n columns, different X can map to one Y.

B. *Symbol*

The symbols used in the figure have the following meanings:

The long and short memory network structure is mainly composed of three parts::

（1）Forget gate layer

The first step in LSTM is to decide what information should be thrown away from the cell state. The decision is made by a sigmoid layer called ‘forget gate layer’. After linearly combining the output information h_{t-1} of the previous layer and the current information X_t, the function value is compressed by the activation function to obtain a threshold value between 0 and 1. The closer the function value is to 1, the more information the memory retains. The closer to 0, the more information the memory loses.[3]

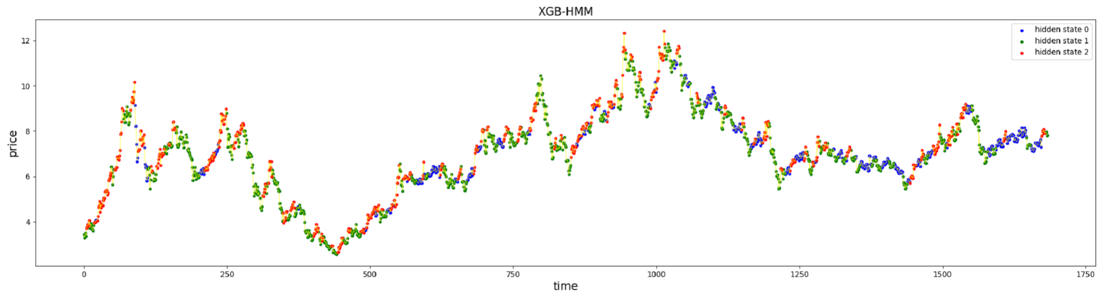


Figure 8. XGB-HMM traning set

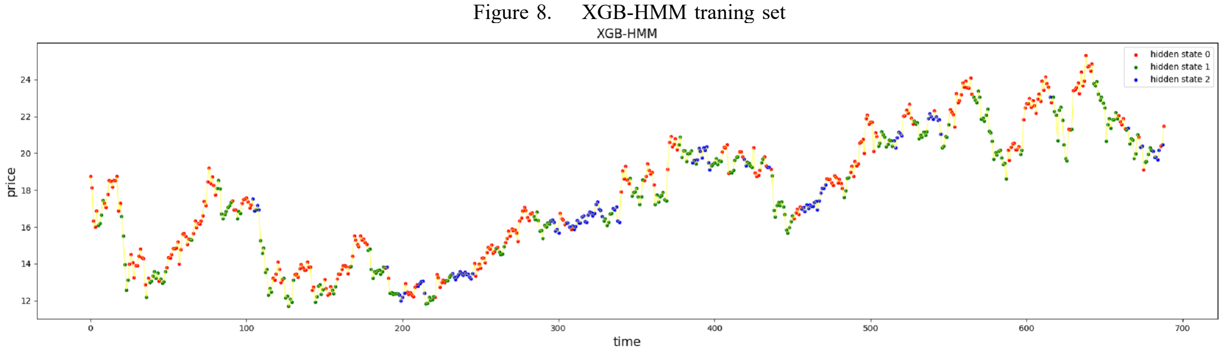


Figure 9. XGB-HMM test set

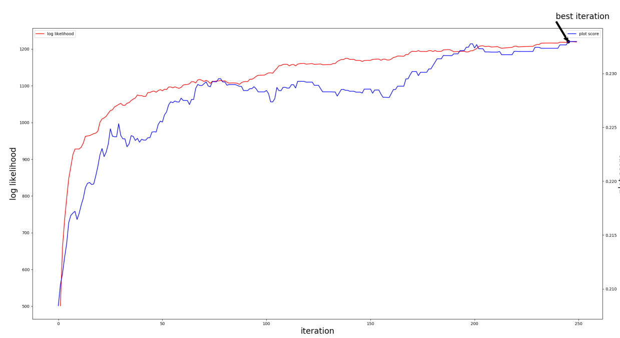


Figure 10. iteration

Symbol	
Type	Meaning
X	Information
+	added information
σ	Sigmoid layer
tanh	tanh layer
h(t-1)	the output of the previous LSTM unit
c(t-1)	memory of the previous LSTM unit
X(t)	current input
c(t)	newly updated memory
h(t)	current output

Table IV
Symbol

（2）Input gate layer The input gate determines how much new information is added to the unit state. There are two steps to accomplish this: First, the sigmoid neural layer of the input gate determines which information needs to be updated; a tanh layer generates a vector that is used as an alternative to update the content C_t. Then, we combine the two parts and update the unit state.

（3）output gate layer Finally, the output gate determines what value to output. This output is mainly dependent on the unit state C_t and also requires a filtered version. First, the sigmoid neural layer determines which part of the information in C_t will be output. Next, C_t passes a tanh layer to assign values between -1 and 1, and then multiplies the output of the tanh layer by the weight calculated by the sigmoid layer as the result of the final output.[3]

C. Costruction of X in GMM-HMM+LSTM

On the previous work,we have done $gmm - hmm_i, i = 1, 2, ..., 8$.

Foe every $gmm - hmm_i, i = 1, 2, ..., 8$， we get one

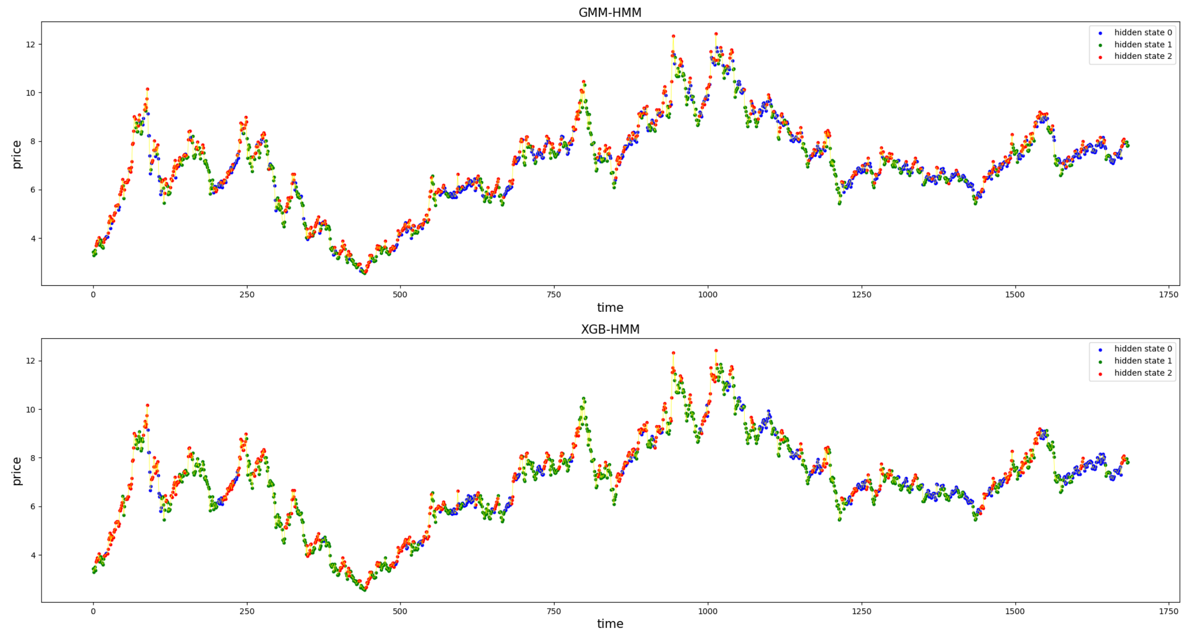


Figure 11. train

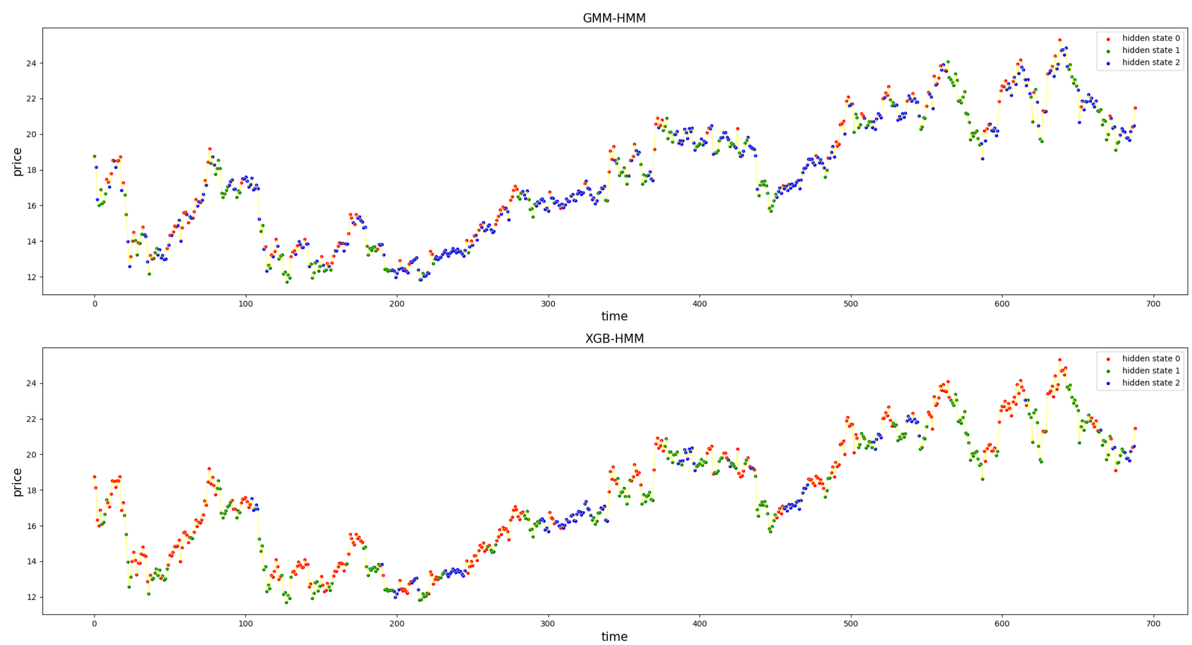


Figure 12. test

$state\_proba, i = 1, 2, ..., 8$，and then put them together to get X。

D. Training algorithm

（1）Run GMM-HMM，and get $P\{S_t = i\}, i = \{1, 2, .., N\}, t = \{0, 1, .., T-1\}$

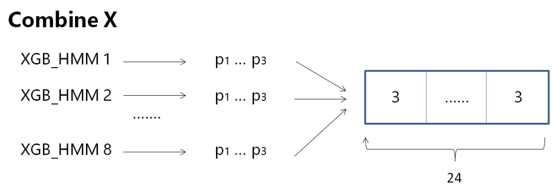


Figure 13. LSTM

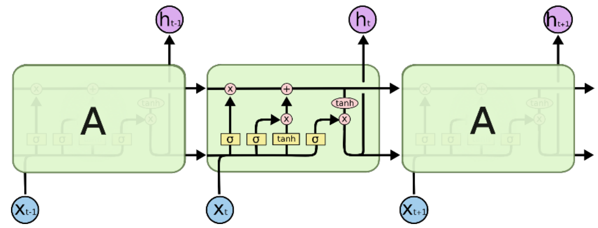
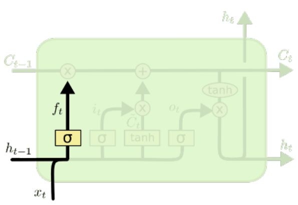
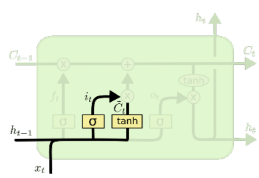


Figure 14. LSTM Process



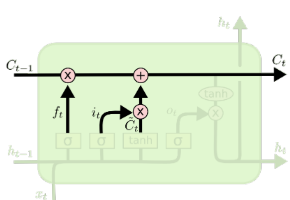
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Figure 15. Forget gate layer[3]



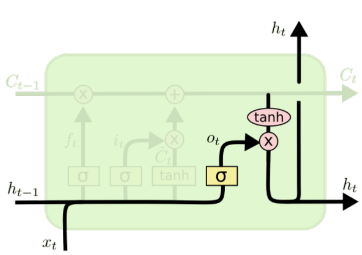
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Figure 16. Input gate layer[3]



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 17. Input gate and candidate gate[3]



$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 18. out put gate layer[3]

（2）Use probability $P\{S_t = i\}, i = \{1, 2, .., N\}, t = \{0, 1, .., T-1\}$ to construct the training set X for LSTM。

$$X = \begin{pmatrix} P\{S_1 = 1\}......P\{S_{T-1} = 1\} \\ P\{S_1 = 2\}......P\{S_{T-1} = 2\} \\ P\{S_1 = 3\}......P\{S_{T-1} = 3\} \end{pmatrix}$$

（3）Input X and Y into LSTM, train LSTM

E. *Results and analysis*

In order to test the effect of our model, we select a number of stocks for model training and determine the various parameters of the model.

For our better understanding, we choose one of the stocks Fengyuan Pharmaceutical (000153.XSHE) which is from 2007-01-04 to 2013-12-17 to do the visualization. The figure is below.

Similarly, we choose one of the stocks from the test data Red Sun data from 2014-12-01 to 2018-05-21 to do the visualization.

The GMM-HMM model is trained on 8 training factors on the training set, and 8 state_proba matrices are generated to obtain X.

Start the training by entering this X into the LSTM model. The model completed in this training is $gmm-hmm+lstm_0$.

On the test set, run the $gmm - hmm + lstm_0$ model and output the result of the lstm model: the accuracy rate is 76.1612738%.

F. *Construction of X in XGB-HMM+LSTM model*

Above, our team has constructed the model $xgb - hmm_i, i = 1, 2, ..., 8$.

For each $xgb - hmm_i, i = 1, 2, ..., 8$, you can get a $state\_proba, i = 1, 2, ..., 8$, and then merge them can get the matrix X.

G. *Training algorithm*

（1）Run XGB-HMM，and get $P\{S_t = i\}, i = \{1, 2, .., N\}, t = \{0, 1, .., T-1\}$

（2）Use probability $P\{S_t = i\}, i = \{1, 2, .., N\}, t = \{0, 1, .., T-1\}$ to construct the training set X for LSTM。

$$X = \begin{pmatrix} P\{S_1 = 1\}......P\{S_{T-1} = 1\} \\ P\{S_1 = 2\}......P\{S_{T-1} = 2\} \\ P\{S_1 = 3\}......P\{S_{T-1} = 3\} \end{pmatrix}$$

（3）Input X and Y into LSTM, train LSTM F

H. *Experimental Results and analysis*

In order to test the effect of our model, we select a number of stocks for model training and determine the various parameters of the model.

For our better understanding, we choose one of the stocks Fengyuan Pharmaceutical (000153.XSHE) which is from 2007-01-04 to 2013-12-17 to do the visualization. The figure is below.

Similarly, we choose one of the stocks from the test data Red Sun data from 2014-12-01 to 2018-05-21 to do the visualization.

At first, we train the XGB-HMM model on the training set with eight types of feature factors respectively and get eight matrix $state\_proba_i, i = 1, 2, .., 8$.

Then ,we merge these eight matrices and get a single martix X.

According to X, we fine-tune LSTM model. The model completed in this training is $xgb - hmm + lstm_0$.

On the test set, run the $xgb - hmm + lstm_0$ model and output the result of the lstm model: the accuracy rate is 80.6991611%..

I. Advantages and Disadvantages and Model Improvement

Advantage:

LSTM is of time series.

In the final task of fitting state_proba->label, we compared the effects of LSTM and XGB and found that LSTM is better than XGB.

Disadvantages:

The accuracy of XGB on the training set is 93%, and the accuracy on the test set is 87%, showing that XGB ourperforms GMM in estimating the emission probability B but there is room for improvement.

Improvements to do：

The processing of the data set and the construction of the features can be made more detailed. Adjust the model parameters to make the final rendering of the model better.

Acknowledgement

We would like to say thanks to MingWen Liu from ShiningMidas Private Fund for his generous help throughout the research. Without his support, it's hard for us to finish such a complicated task.

Reference List

[1] Dong Yu, Li Deng, Automatic Speech Recognition_ A Deep Learning Approach , Springer-Verlag London, 2015.

[2] Silver D, Huang A, Maddison C J, et al. Mastering the game of Go with deep neural networks and tree search[J]. nature, 2016, 529(7587): 484.

[3] Christopher Olah, 朱小虎 Neil, https://www.jianshu.com/p/9dc9f41f0b29.

[4] huangyongye, https://www.cnblogs.com/mfryf/p/7904017.html.

[5] Marcos Lopez De Prado, Advances in Financial Machine Learning, Wiley, 2018, 47.

[6] 李航，统计学习方法，清华大学出版社，2012，第八章.

[7] Gers F A, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM[J]. 1999..

[8] Sundermeyer M, Schlüter R, Ney H. LSTM neural networks for language modeling[C]//Thirteenth annual conference of the international speech communication association. 2012.

[9] Chen T, Guestrin C. Xgboost: A scalable tree boosting system[C]//Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016: 785-794.

[10] Rabiner L R. A tutorial on hidden Markov models and selected applications in speech recognition[J]. Proceedings of the IEEE, 1989, 77(2): 257-286.

[11] Hinton G, Deng L, Yu D, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups[J]. IEEE Signal processing magazine, 2012, 29(6): 82-97.