**Introduction:**

This project aims at carrying out stock market analysis using various data mining machine learning methods in order to help investors maximize their profit during stock investment.

First of all, Hidden Markov Model (HMM) is adapted as a representative of ‘pure computer science’ method that relies only on close data of each day, without heuristics from financial or business industry. After fitting a discrete HMM or continuous HMM with Gaussian Mixture Model (HMM-GMM), we can do 2-class classification(up/down) or regression(prediction on close price), correspondingly.

Secondly, Support Vector Machine(SVM) is used as classifiers that employ industrial heuristics – stock indicators.

**Data Preprocessing:**

Both methods use the data source that been preprocessed in following way:

1. Calculate all indicators (as listed in SVM method section) from original data.

2. Normalize each tuple (open, close, volumn, indicator values etc…) of each stock to be mean = 0 and standard deviation = 1 to eliminate inter-stock, inter-tuple and inter-indicator variance.

**HMM:**

Under the circumstance of time-sequence modeling, HMM hold assumption that p(n+1) is dependent on p(n), which determines a markov chain, to simulate a time-sequence model. Compared with assumption made by bayes model, which grant the independence between p(n+1) and p(n) (under the circumstance of time-sequence modeling), I want to make the hypothesis that the dependence assumption by HMM is more reliable, according to my intuition. So I prefer HMM (discrete and continuous) rather than bayesian classification for fitting time-series stock data.

**Basic operation:**

HMM train:

In order to fit an HMM model(no matter discrete or continuous), a series of data need to be provided.

Parameter:

O: observation, number of features, in this case, o=1 stands for only use close value.

S: number of hidden state, tunable

T: window size, tunable

Prior probability vector,Transition matrix and emission(observation) matrix are randomly initialized using o and s.

Continuous HMM train:

Apart from parameters above, emission matrix can not be initialized because it follows the Gaussian mixture distribution(GMD). Additional parameter need to be provided for fitting such a distribution:

M: number of Gaussian mixture

So that the GMM will be modeled and further be used to model continuous HMM.

Finally, HMM is trained using EM (Baum Welch) algorithm.

HMM classification/regression

HMM don’t provide directly provide classification/regression result. Instead, it provide loglike probability for each testing sequence. We can do classification according to its argmax (max likelihood).

For example, under a test case of window size w=5, known sequence s=[1 2 3 4] and regression value domain [1,10], classification label domain[1,2,3,..,9,10].

Classification analysis should be:

Loglik[1] = loglik(DHMM,[s 1])

Loglik[2] = loglik(DHMM,[s 2])

…

Loglik[10] = loglik(DHMM,[s 10])

result = argmax(loglik)

Regression is derived from classification, with a fixed discretizer for to transfer continuous data to categorical (discrete) data.

Regression analysis should be:

Discretizer = 100;

Testvalues = discretize(value domain,discretizer)

For all testvalues

Loglik[n]=loglik(MHMM,[s testvalues[n]])

result = argmax(loglik)

**Preprocessing**

Apart from preprocessing specified before for normalization, we also need to split training/testing dataset and prepare data for classification/regression.

Split training/test set:

For all data records on 1800+ transaction days, the training and test set are split at 80% percentile according to date, which maintains the continuity.

Training/test accuracy:

For both training and test data, we will classify or regress the corresponding attribute of each day, assuming the previous t days records are known (from training and testing set), which is identical to practical situation. The detailed algorithm is to adapt HMM classification/regression on a moving window with step size 1 on corresponding dataset.

Experiment 1: Discrete HMM, binary classifcation.

Input: binary sequence which stands for stock going up/down compared with close price of previous day.

Observation: O = 2 (binary, 0 and 1)

Tunable parameter:

Window size: T = 5

Number of state: S = 15

Accuracy:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Stock | Stock1 | Stock11 | Stock13 | Stock23 | Stock293 | Stock857 |
| acc\_train | 0. 5828 | 0.6003 | 0.5763 | 0.5681 | 0.5893 | 0.5849 |
| acc\_test | 0.4919 | 0.5368 | 0.5259 | 0.5014 | 0.5177 | 0.5226 |

Experiment 2: Discrete HMM, discretized classification to simulate regression.

Input: discretized earn value:

Earn(t) = (close(t)-close(t-1))/close(t-1)

Reason of failure:

According to statistical plotting, we can find that the input sequence value roughly forllows gaussian distribution. So that the frequency of the label stands for the mean of the input data is dominating. So the classification result will be highly biased that the average label will be dominant. So that all our regression result using discrete HMM will be the average earn value.

Experiment 3: Continunous HMM, sequence regression

Input: normalized close value where mean = 0 and standard deviation = 1. Split dataset to training set (80%) and test set (20%)

Parameters:

Number of observation O=1, only the close value.

Tunable parameters:

To get the best combination of [s,m,t] ([number of hidden state, number of mixture in GMM, window size]), a number of combinations are tested under the close value of stock 1 and evaluation methods below being carried out. Finally I pick up few combinations:

S = 35(fast)/ 50(accuratez)

M = 15

T = 3

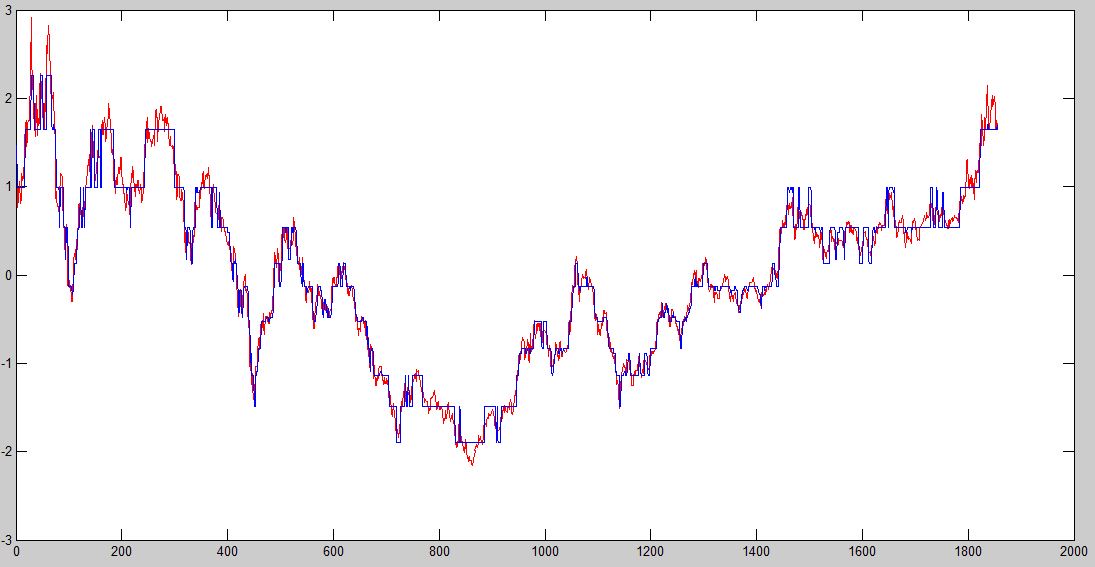
Discretizer on prediction = 100 (100 partition of domain value)

For T=3, prediction is based on previous 2 days.

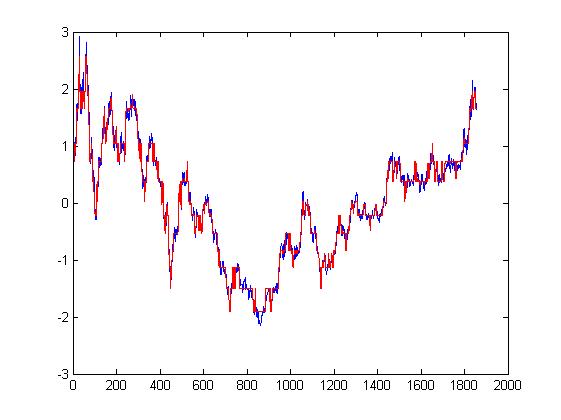
Result by MSE:

S=35;M=15,T=3 (result.mat)

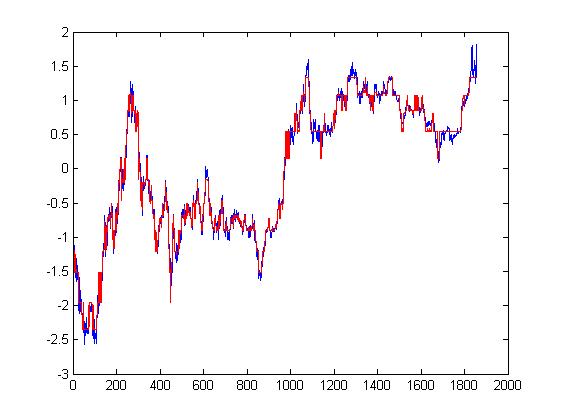
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stock | Stock1 | Stock11 | Stock13 | Stock23 | Stock293 |
| acc\_train | 0. 2227 | 0.1689 | 0.1502 | 0.0853 | 0.2059 |
| acc\_test | 0.1257 | 0.1985 | 0.1207 | 1.5197 | 0.6257 |

stock 1 S=50, M=15, T=3

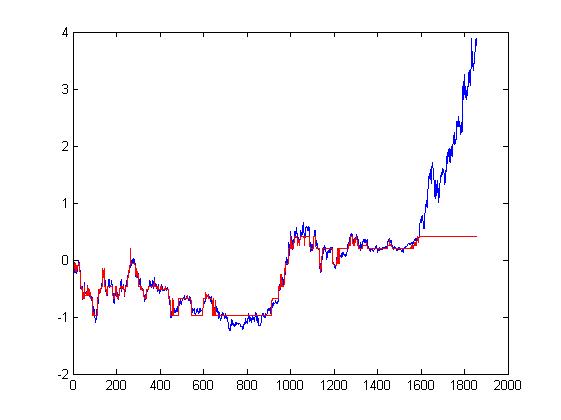
stock 1 S=50, M=10, T=3 (res5010.mat) err\_train:0.1345 err\_test:0.1138



stock 11 S=50, M=10, T=3 (res5010.mat) err\_train: 0.1253 err\_test:0.1138



stock 23 S=50, M=10, T=3 (res5010.mat) err\_train:0.0885 err\_test:1.5197



**SVM:**

Introduction:

The mechanism of SVM is basically to separate different classes using hyperplanes, which are determined most likely by support vectors that on the borders on classes. SVM need a input of n-dimensional features and class labels to determine the inter-class boundaries.

Features:

The features of SVM training and testing are indicators that been calculated from raw data (open, close, high, low, volume). There are totally 16 features and they are listed below:

WILLR, ROCR3/12, MOM1/3, EMA6/12, MACD, OBV, RSI6/12, ATR14, MFI14, CCI12/20, TRIX

Labels:

The label is binary (0 and 1) which indicates the trend of close value of next transaction day, compared with close value of today:

The label ensures that, in real case, the features(indicators) of transaction day p is calculated from previous days but label(p) is unseen.

Training/Test data seperation

After calculating features and labels, all records of each transaction day are considered as samples, rather than time series. So the sepration of training and test set is a randomsplit of original data source.

Dimension reduction – PCA

Since there are totally 16 features with only no more than 1850 records available, the SVM won’t converge in limited iterations practically, which requires reduction on feature dimensions.

In this experiment, we choose first 8 eigenvectors for dimension reduction to represent at least 90% of original feature space. Correspondingly they are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Stock | Stock1 | Stock11 | Stock13 | Stock23 | Stock293 | Stock857 |
| Variance | 0.9063 | 0.9027 | 0.9118 | 0.9124 | 0.9035 | 0.8996 |

Testing accuracy:

Five experiments, totally different training/testing split and svm models, are done for each stock data to eliminate random cases, the averaged accuracy shows:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Stock | Stock1 | Stock11 | Stock13 | Stock23 | Stock293 | Stock857 |
| accuracy | 0.5285 | 0.6860 | 0.7115 | 0.6851 | 0.6813 | 0.6918 |