CS7646 Machine Learning for Trading MC3_P3 Report
Yuzhou Li
903145989

I. Introduction
There are 5 parts in this project report. For part II, I implemented 3 technical indicators, which are Bollinger Band, n-day-momentum, moving average convergence divergence (MACD). In part III, I created a rule based trader by using the three technical indicators. Part IV is a machine learning based trader, which contains a random tree based bagging learner. Part V focuses on comparing performance of rule based trader and machine learning based learner with benchmark. Here, the definition of benchmark is holding 500 shares of IBM stocks and compare portfolio value with our portfolio value.

II. Implement of Technical Indicators
The first indicator is momentum. Momentum describes the trend of a stock price. Usually, 10-days momentum is a good indicator, which compares the current stock price with that 10 days later. In order to make it like a derivative, I will compute 2-days-momentum and use it with Bollinger Band. Following figure shows a normal 10-days-momentum:

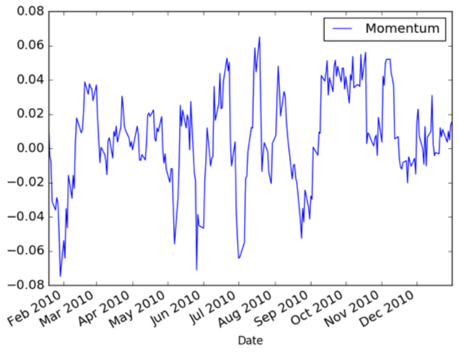


Fig. 1 IBM (2010.01.01-2010.12.31) Momentum

Bollinger Band is more complex. It can measure the "highness" or "lowness" of the price relative to previous trades. %b is a good indicator, where %b = (last − lowerBB) / (upperBB − lowerBB). In my code, it has the column name BollingerBand.

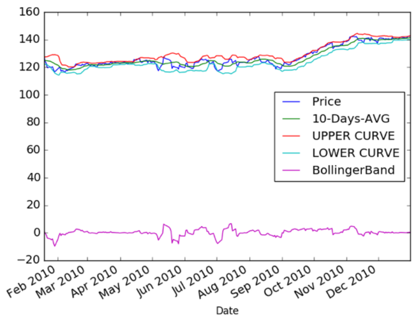


Fig.2 IBM (2010.01.01-2010.12.31) Bollinger Band

The last indicator is moving average convergence divergence (MACD). In MACD, exponential moving average (EMA) will be used. Exponential moving averages highlight recent changes in a stock's price. By comparing EMAs of different lengths, the MACD series gauges changes in the trend of a stock. The difference between the MACD series and its average is claimed to reveal subtle shifts in the strength and direction of a stock's trend. It may be necessary to correlate the signals with the MACD to indicators like RSI power. In order to calculate that, following equations will be used:

$$DIF = EMA_{(close,12)} - EMA_{(close,26)}$$
$$DEM = EMA_{(DIF,9)}$$
$$OSC = DIF - DEM$$

When DIF curve is going to across EMA curve from below then it is a strong signal to buy. When DIF is going to across EMA curve from above, then it is a strong signal to sell. The MACD can be illustrated as below:

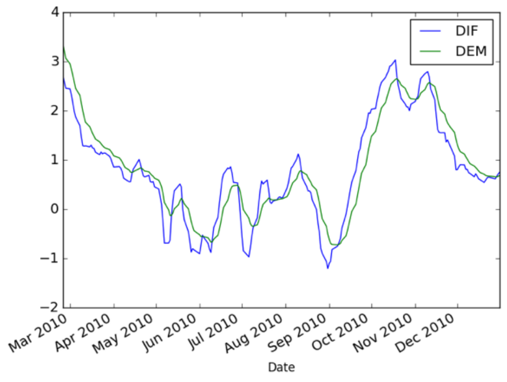


Fig.3 IBM (2010.01.01-2010.12.31) MACD - DIF & DEM Curve

III. Rule Based Trader

(BLACK LINES IN FOLLOWING FIGURES MAY OVERLAP WITH OTHER LINGS)

In order to implement rule based trader, we need to define when to buy and when to sell. Bollinger Band is a good indicator and it need the help of current trend. Here, I use 2-day-momentum to indicate the current trend. This is a good substitute. 2-day-momentum itself can also be a weak indicator. If Bollinger Band condition does not meet, then this weak indicator can also perform. As for MACD, it is a mid or long term indicator. For the first 26+9 trading days, it will not work, so in my code, orders will only be executed after that period. I personally believe MACD is a medium indicator, which is better than 2-day-momentum and worse than Bollinger Band.
In order to combine all indicators and make decision, I create a variable *con*. I calculated all technical indicators on a certain day, and allocate different weight to every indicator. Variable *con* is the result of the weighted indicators. **I set several conditions, and readers can find them in my code rule_based.py function cal_conservative(). Basic idea is if BB buying condition meets, then variable gets 0.8 weight. If buying momentum**

condition meet, plus 0.2. If buying MACD conditions meet, plus 0.5 or 1. If the condition is about selling, then all weights are negative. If the final con is 0, then use MACD OSC value with some constant to generate a *con* value. All conditions are based on my aforementioned assumptions, and were tested a lot of times. If variable *con* is equal or greater than 0.6, then system will execute LONG. If variable *con* is less or equal than -0.6, the system will execute SHORT. **Due to the market simulator was designed to execute BUY and SELL, I converted LONG into a pair of BUY and SELL, and SHORT a pair of SELL and BUY.** The result of IBM from time period 2006.01.01-2009.12.31 is shown below:

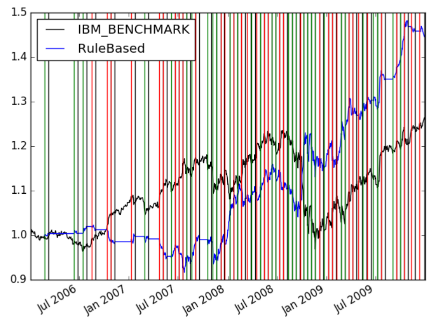


Fig.4 IBM (2006.01.01-2009.12.31) Rule Based Trader Result (INSAMPLE)

As for the out-sample data, time period was from 2010.01.01-2010.12.31. Frankly, the result was not good enough and result is shown below:

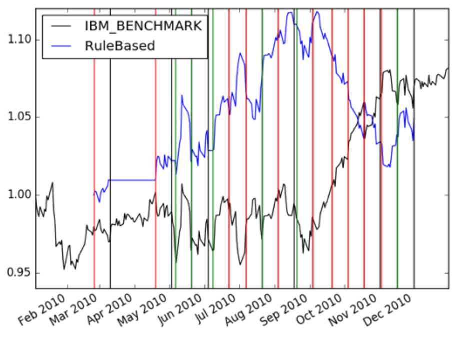


Fig.5 IBM (2010.01.01-2010.12.31) Rule Based Trader Result (OUTSAMPLE)

The auto trader can still make some money in this period but it failed to beat the benchmark.

IV. Machine Learning Based Trader

In this part, **I used the random tree classifier rather than a random tree regression learner. I modified the random tree file built in MC3P1 and make it return statistical mode rather than statistical mean value.** The learner I used was a bagging learner, which contains a number of random tree classifier. I picked current price, 10-day average price, BB upper band, BB lower band, %b, 10-day momentum, 2-day momentum, DIF, DEM, OSC as the training data column and **check if the price of a stock 10 days after that day is higher than 1% or lower than 1%. If it is higher, then label training Y value as +1, if it is lower than 1% than Y is -1; all other condition will be 0.** According to my test, if I set the percentage much higher than 1%, i.e. 10%, there will be a lot of 0 in training data Y. Under that condition, there will be only several BUY and SELL within one year and the accumulative return will perform very bad.

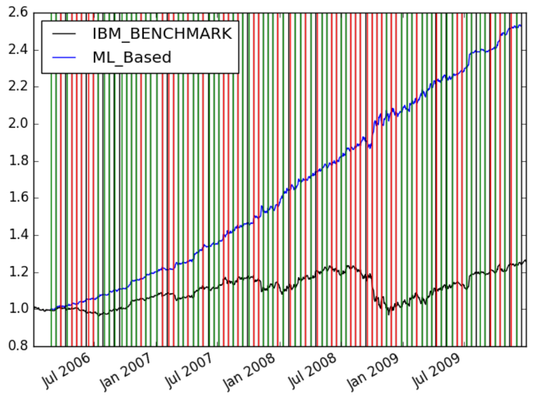


Fig.6 IBM (2006.01.01-2009.12.31) ML_Trader Performance (leaf:5, bag:20)

The above figure performs very well, and this is because the training data and test data are the same. If I use the same training data to predict out sample, the result is:

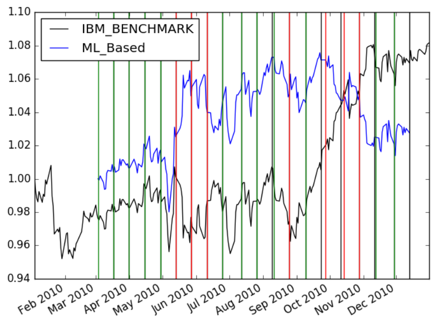
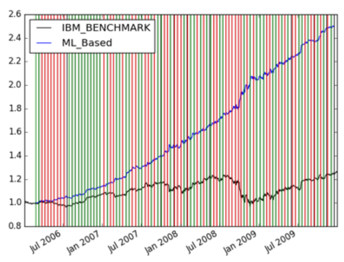
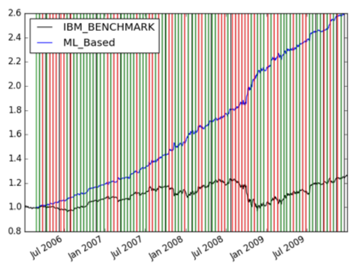
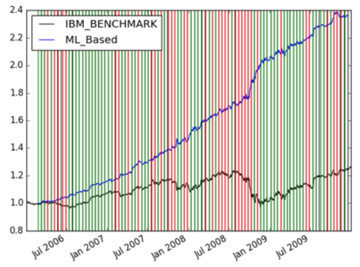


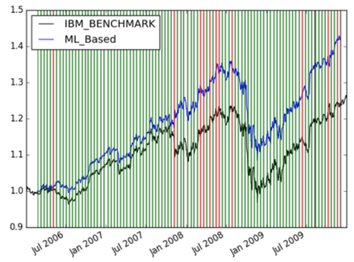
Fig.7 IBM (2010.01.01-2010.12.31) ML_Based Trader Result (leaf:5, bag:20)

The following figures shows the parameter relationship.









Leaf-size:5, Bag: 10	Leaf-size:5, Bag: 100
Leaf-size:10, Bag:20	Leaf-size:50, Bag:20

Fig.8 IBM In-Sample Data ML_Trader Parameter Test

From the above chart we can conclude the smaller leaf-size and larger bag number will cause better in-sample performance, and this is a kind of over-fitting.

V. Comparative Analysis

For the rule based trader, out sample test result in Fig.5. For the ML based trader, Fig.7 is the result. In order to get better performance, we could increase the leaf size and this will help us to avoid over-fitting.

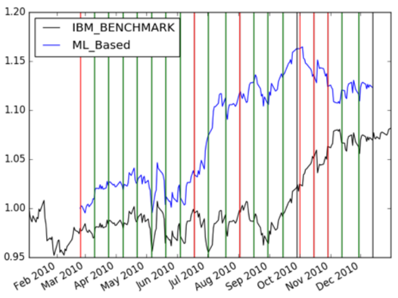


Fig. 9 ML Based Trader Out-Sample Result (Left: lf=100, bg=50)

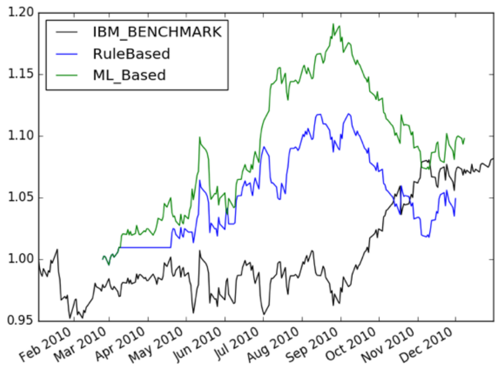


Fig. 10 Out-Sample Trading Comparison (lf:90, bg:20)

From the above figure we can conclude that ML trader sometimes can provide opposite Long/Short signal compared with rule based trader. (See Mid August period) Also, this rule based trader is much more conservative because it has higher opportunity to do nothing. Meanwhile, the ML trader is not stable, which means every time it will produce different result. Statistical comparison values are shown below:

Method	IBM_Benchmark	Rule_Based Trader	ML_Based Trader
Highest In-Sample	1.25	1.45	2.6
Highest Out-Sample	1.08	1.05	1.13

For the rule based trader, all decisions are made by given rules, which is related highly depend on technical indicators. As for ML based trader, it has to learn everything by itself. The condition of triggering LONG or SHORT is different. ML trader will learn it highly depend on the stock price 10 days later while the rule based trader will judge it by given rules. As a matter of fact, ML trader will definitely have a better in sample performance and if we could use 100% over-fitting that means leaf size equals 1, and bag number is large enough, we could get a "never lose money" in sample result.
For out sample test, things are different. The rule based trader performs not very well because the given rules are not good enough. Meanwhile, if we could avoid over-fitting, the ML trader will perform better but not a lot better. Sometimes, ML trader will lose more money than rule based trader. So, it seems that rule based trader is more stable.