Project 4 Report **Chao Shi**
CS7646, Summer 2016 GTID#: 902694979

1 Overview

1.1 Features and Learning Algorithm

I use KNN as learner (k = 3) in the code. The predicted Y value is the average

The features I chose are (for day t≥5):

- Momentum of today w.r.t 5 days' ago: price[t]/price[t-5]
- Today's price normalized by (twice) Bollinger Bands: (price[t] - sma[t])/(4*stdev[t])
- Volatility normalized by mean: stdev[t]/sma[t]*10

Y was given: price[t+5]/price[t] - 1, i.e. The momentum of the price 5 days from today w.r.t today.

Since we're dealing with time series data, we can't add training data all at once into the KNN learner as we did in project 2, otherwise we would be able to see the future in training. Thus we can only add data before today (day t) into learner to predict Y.

1.2 Trading Policy

As said in the project statement, making a good policy is hard. I started with an easy (intuitive one) and then see where it goes. In the end I have three versions of trading policy:

- Naive version: if predicted Y > 0, i.e. price[t+5] > price[t], we enter or stay in long position (+100 shares) else we enter or stay in short position(-100 shares). There's no middle state (holding zero shares) once we started.
- Simple version: MACD oscillator. Specifically, 10 day EMA and 40 day EMA to construct signal line. If EMA10 just crossed EMA40 from above, we enter or stay in short position. If EMA10 crossed EMA40 from below, we enter or stay in long position.
- Complex Version: Utilizing the local minimal/maximal together with EMA40 to determine buy or sell. If EMA10[t] > EMA40[t] and EMA10[t-2]<EMA10[t-1]>EMA10[t], it means the price has already just passed the peak starting to fall back, and this is a strong sell/short signal. On the contrary, a local minimal of EMA10 below EMA40 is a sign of price the bouncing back, which is a strong buy signal. I chose EMA instead of SMA because EMA is more weighted towards recent trend, thus has less lagging than SMA, which is important to predict the price trend.

The cruel fact is that NONE of the above policy really works on real world data, but I did figured out where the problem probably is and have some thoughts about improving it. So guess it's not a total failure.

2 Results

2.1 SINE data

As mentioned in first Naive policy works well on SINE data.

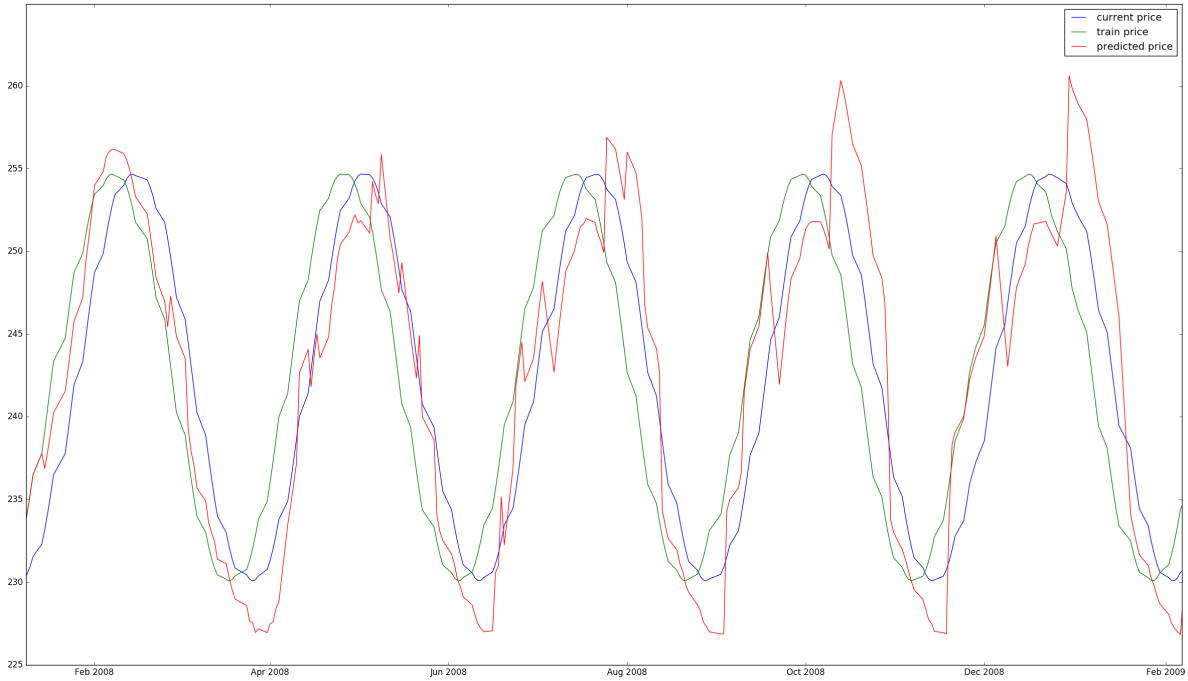

Figure 1: Price prediction

Figure 1 shows the predicted price, corresponding training price and current price. The former two were shifted back 5 days. We can see that the prediction (red) is not perfectly periodic in the beginning. This is because each X is trained on different data. Initially there were only a few data points thus any new added would affect the prediction significantly, while later there're already many training points added to learner, the effect of adding a new point is greatly reduced. Thus the prediction starts to look almost perfectly periodic as we can see from the last two peaks.

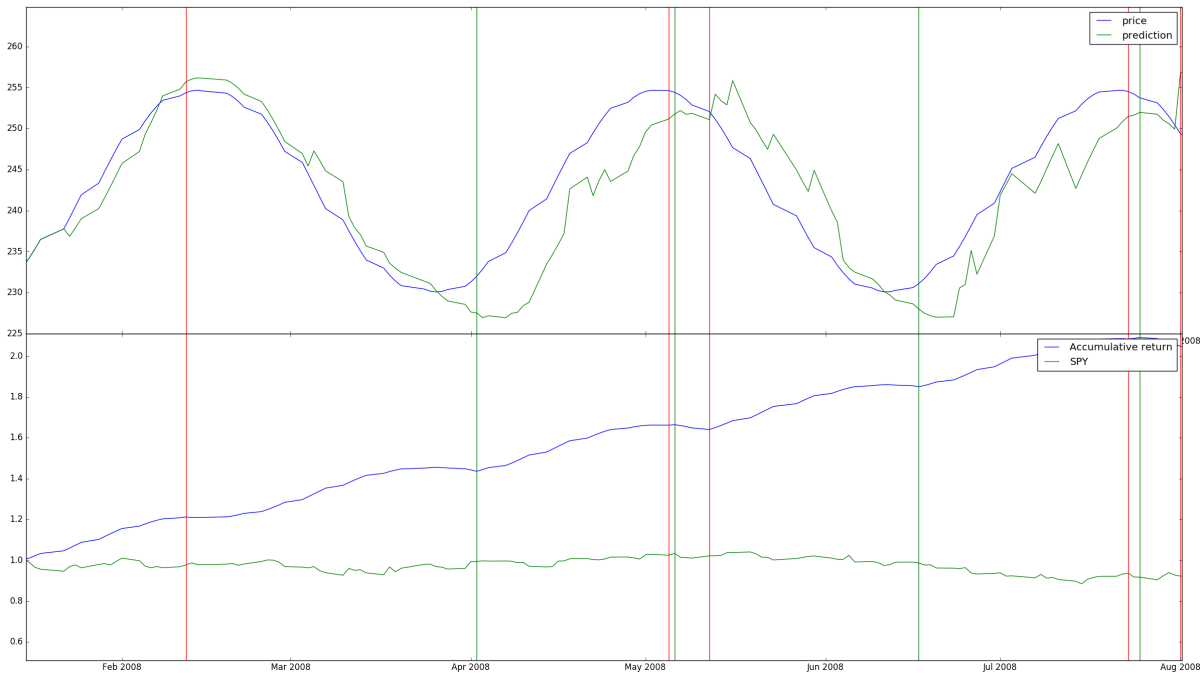

Figure 2: In sample backtest

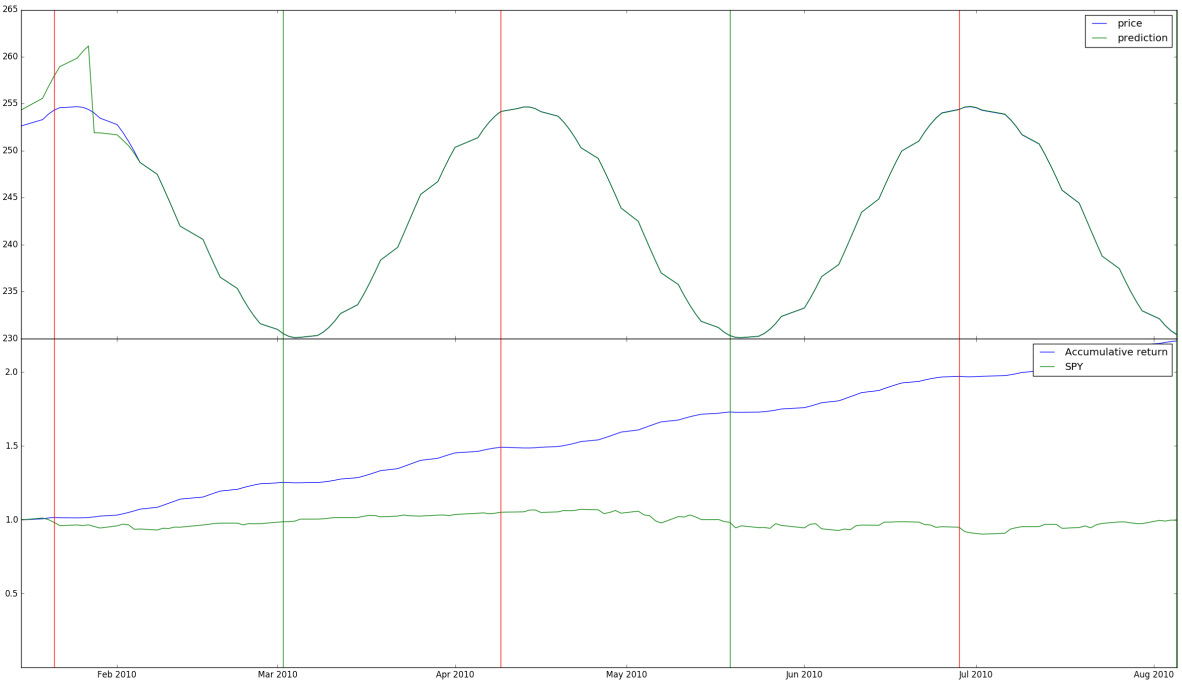

Figure 3: Out sample backtest

Figure 2 and 3 shows the in and out sample back test on the given time range.
Upper panel of each figure is the comparison of predicted price and true price with trading dates marked by horizontal line. Red means entering short position, green means long. For in sample

test, we see that although it doesn't enter long/short position in the absolutely best point, it's very close. The error comes from the interpolating nature of KNN and what we predicted is what's happening 5 days from now, there is a lagging effect. For out sample test however, the prediction perfectly match the real price except for the initial discrepancy.
Both of the accumulative return is near optimal, holding long position when the price goes up, short position when the price goes down.

2.2 IBM data

First look at the prediction in figure 4:

Figure 4: IBM prediction

Again the red line is the predicted one. We can see that although the general trend of prediction is closely correlated with the true price, it's lagging behind the true price of a roughly a constant amount of time (about 8 days), and it gets messier at the tail, correlations seems going off significantly. I think the lagging effect comes from the averaging nature of KNN. The trend of prediction is bound to be slower if it's obtained by a form of averaging. Linear regression also has this problem. After all linear regression is also a form of average, the average of slope. This is also why I gave up LinReg because it seems almost any features I can think of is not linearly correlated to the target, thus the result of linear regression is severely off. The lagging of prediction and jittery of price basically sets the tone that we can't rely on the price prediction alone to make trading policy.

It doesn't hurt to try however, so Figure 5 and 6 are the in and out sample back test based on the same naive policy works well on SINE data.

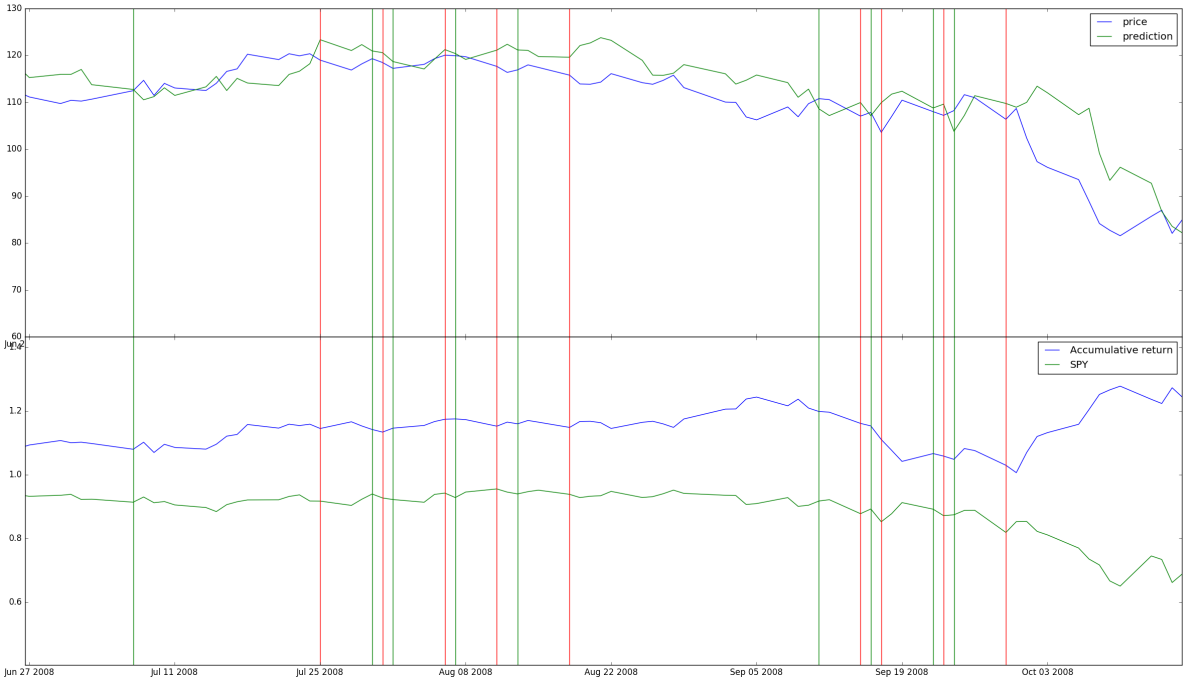

Figure 5: IBM in sample back test

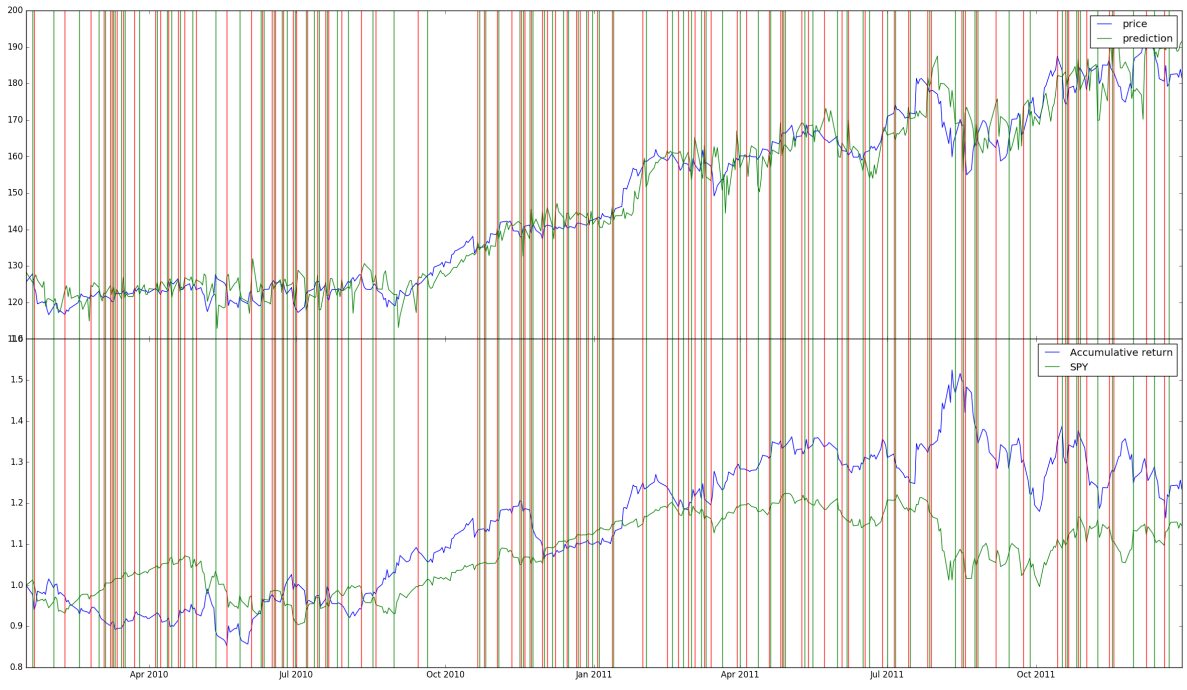

Figure 6: IBM out sample backtest

Still for both figures upper panels are price chart with trading days marked and lower panels are portfolio performance in accumulative return. We see from both figures that when the trading days are sparse, the performance is more correlated to price trend which makes

sense because sparse trading days means less turn of trends given our strategy, which further means the line is less noisy and can actually trending for a little while, and these information were captured by our linear model. Unfortunately most of the time the line is relatively noisy so that we make enough seemly random trading decision to make the performance follow the trend of price , as we can see from figure 6 clearly. After so many almost randomly invoked transaction the accumulative return has roughly followed the market.

2.3 Improvement

So naive approach doesn't work well on IBM data, we simply can't use predict price alone. Prices are noisy, so we smooth it. That's one reason why we use SMA, EMA, etc. In Figure 7 below we show the in sample test of the simple approach outlined in section 1, that is using MACD alone.

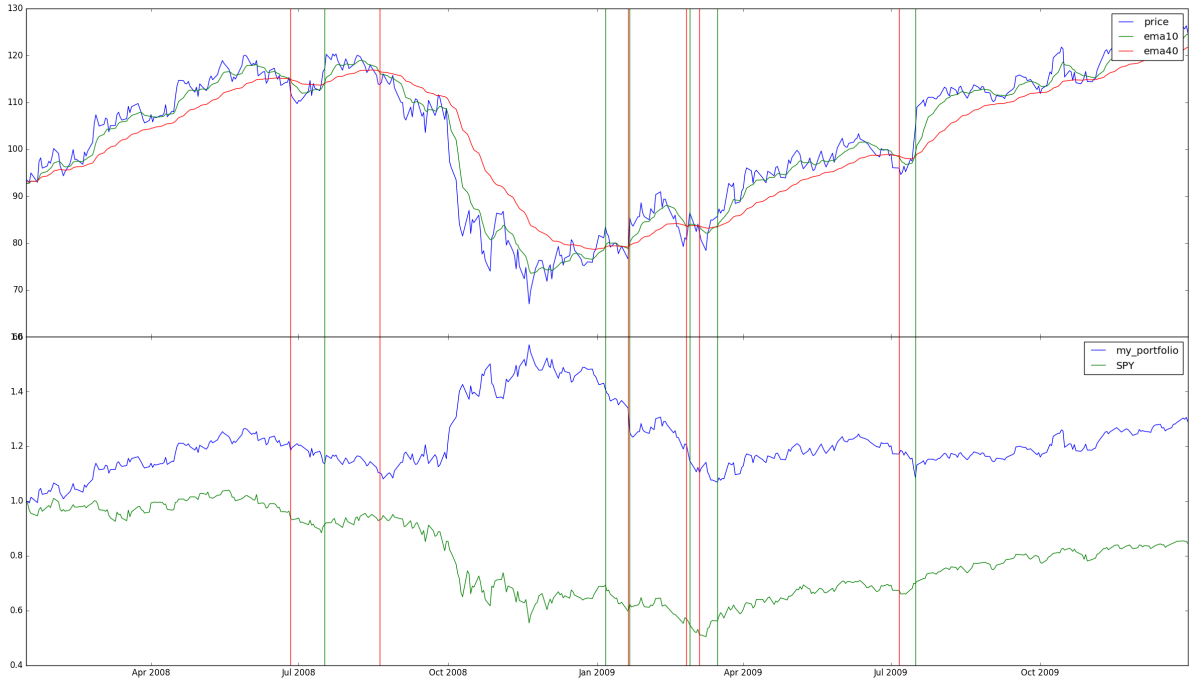

Figure 7: MACD in sample back test

First we see that the accumulative return looks OK, although I expected better given how well the MACD oscillator captured the overall trend of the stock price. It can't be seen in this figure but we entered the long position immediately on day 1 given the predicted uptrend of EMA40 calculated from predicted price. This is actually the only place we used the predicted price here. Then it is obvious that the trading decision did exploit the information from MACD to capture the big up and down trend in price, correspond directly to the rise of accumulative return in the beginning up trend, middle big down trend and tailing up trend.

However, the two year return is still just 30%, although it was as high as close to 50% after the first year, it fell back. The reason is that during the transition of big up and down trend, the stock price is temporarily highly oscillating, not trending, thus very noisy. As we can see from the figure there's a stride of dense trading mark to the right of middle, although just several seemingly random transaction, it has done significantly damage to the performance.

This exposes one severe weakness of MACD oscillator that it relies too much on trending. A little transition period could fooled it.

Then I went further trying to get rid of the random transaction we don't want. The first part of it is mentioned in section one already, using local minima/maxima of EMA10 together with EMA40. Also I made a rule that every sell price has to be higher than the previous buy price if the sell is triggered by the maxima of EMA10. I hope the money-losing random transaction can go away with this. It actually does. But not just them, almost all other beneficial transaction are blocked. While I was playing with the price difference parameter, once I set it to 0.01 all the transaction are gone except the buy in the first day. Ironically, this, is actually the best performance for both in and out sample test. Yes, just buy the stock and leave it there it will make more money than my effort trying to "optimize" it does.

3 Further Thoughts

I think the main problem now boils down to how to deal with the fast-oscillating and non-trending phase of a stock. I think Bollinger band trading can definitely help that, but I'm running out of time before I can submit this so here's that.
To be continued...