MC3 – Project 2

Data Generation Logic with Technical Indicators

In order to create data for learner, three technical stimulators were selected to calculate for the training data using the price history. Those technical indicators are:

- **Bollinger Bands**
- **Momentum**
- **Volatility**

Each technical indicator will take the pricing data within a period of time and transform them into a new data that will be used to indicate the buying and selling signal. A window size is also used to calculate the indicator using pricing data. The logic to create training features for the learner was implemented as follow:

- **Bollinger Bands:**
  - Get the price data within the provided time from the data source (csv file).
  - Calculate the mean of the price data within the window size
  - Calculate the standard deviation (std) of the price data within the window size.
  - Create a Bollinger Band's value using the following equation:
    - ***bb_value = (price – mean)/(2*std)***
  - The logic will be repeated until all of the pricing data was used to calculate the Bollinger Band's value.
  - The final values is a list of Bollinger Band's value for each

- **Momentum:**
  - Get the price data within the provided time from the data source (csv file).
  - For each data item, calculate the momentum's value by taking differences between the data item and the future data item within the window size. The detailed equation to calculate for the momentum is:
    - ***momentum_value = (price – future_price)/future_price***
    - ***future_price = price[t + windows]***
    - ***t is the index at the current iteration.***
  - The logic will be repeated until all of the pricing data was used.

- **Volatility**
  - Get the price data within the provided time from the data source (csv file).
  - Calculate the daily return for each pricing history
  - Calculate the standard deviation of the daily return within a provided window.
  - Repeat the process until all pricing data was used.
  - The volatility's value is the set of calculated standard deviation of the daily return.

After the three technical indicators was implemented successfully, we can run the pricing the data from 2007 to 2009 into each implementation to generate three set of indicators value. An array will be used to store all three set of indicators' values. This array will be used to the learner as the Training X.

The Learner that will be used in this project is Linear Regression Learner.

The Training Y that will be used in the learner is the future 5-days return. It can be calculated using the following equation:

- ***return[t] = (price[t+5]/price[t]) – 1***
- ***t is the index of the current iteration***

With the Training Y and Training X, the learner now can be used to predict the 5 days return based on the pricing data.

Trading Policy

After getting a prediction, a set of policy will go through the prediction to make a decision on which action that needs to take in order to maximize the portfolio profit. The policy was implemented as the following:

1. The policy will go through each date on the Predicted data.
2. If the predicted 5-day returns at the current date index is higher than 1%, do following action:
    a. If the portfolio is not in the Short position, the portfolio will enter the Long Position and the action that it will need to take is **buying 100 stocks** in the current date index. If the portfolio is already in the Long position, it will increase the amount of holding stock by **buying another 100 stocks**.
    b. If the portfolio is in the Short position, the action that it will need to take is **buying all shorting stocks** in order for the portfolio to exit the Short position and return it to Neutral position.
3. If the predicted 5-day returns at the current date index is less than -1%, do the following action:
    a. If the portfolio is not in the Long position, the portfolio will enter the Short position by **selling 100 stocks** at the current date index. If the portfolio is already in the Short position, it will increase the amount of shorting stock by **selling another 100 stocks.**
    b. If the portfolio is in the Long position, the portfolio will exit the Long position by **selling all the stock** that it holds and reset its position to Neutral.
4. If the predicted 5-days returns is between -1% and 1%, do the following action:
    a. If the portfolio is not in neither Short or Long positions, the portfolio will count how many days that it maintains the position and increase it by 1.
    b. If the portfolio is in Short position and the day that it maintains the position is more than 5 days, the portfolio will exit position by **buying all shorting stocks** and reset its position to Neutral.
    c. If the portfolio is in Long position and the day that it maintains the position is more than 5 days, the portfolio will exit position by **selling all holding stocks** and reset its position to Neutral.

A portfolio's order file will be generated when the predicted data runs through the policy in order for it to be used in the market simulator in the next steps.

ML4T_399 Training Learner

In this step, the program will take a training dataset, generate a Training X and Training Y for the linear regression, and test the learner again with Training X in order to see how well the learners performs. Here are some features that I used to run the training process:

- Data: **ML4T_399**
- Training Starting Date: **2007-12-31**
- Training Ending Date: **2009-12-31**
- Testing Ending Date: **2009-12-31**
- Testing Ending Date: **2010-12-31**
- Learner: **Linear Regression**
- Technical Indicators: **Bollinger Bands, Momentum, Volatility**
- Window's Size: **12**
- Portfolio's Starting Value: **$10,000**

Training Y/Price/Predicted Y Chart

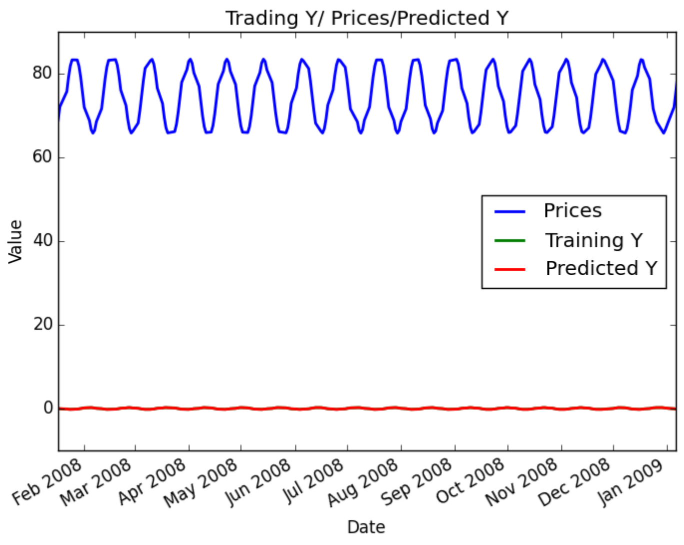

Chart 1 – Trading Y/ Prices/Predicted Y Chart with a zoom into 2008-year period.

Since the predicted Y was following the training Y strictly and their value is very low comparing to the Prices, I generate another charts that zoom into the Training Y and Predicted Y to see how they related to each other.

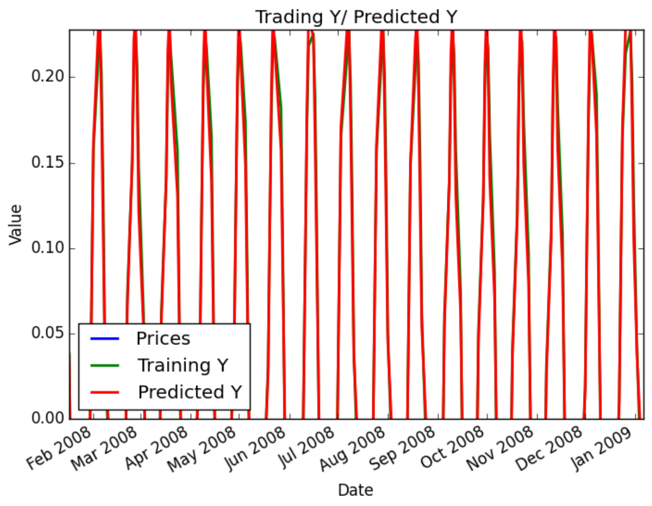

Chart 2 – Trading Y/Predicted Y Chart with a zoom into 2008-year period.

As we can see in the chart, since we use the Training X to query for the Predict Y. The Training Y followed the Predict Y strictly during the Training process.

In the next step, the trading policy will be applied into the Predict Y to see how we can use the predict value to create ordering actions.

Sine Data In Sample Entries/Exits Charts:

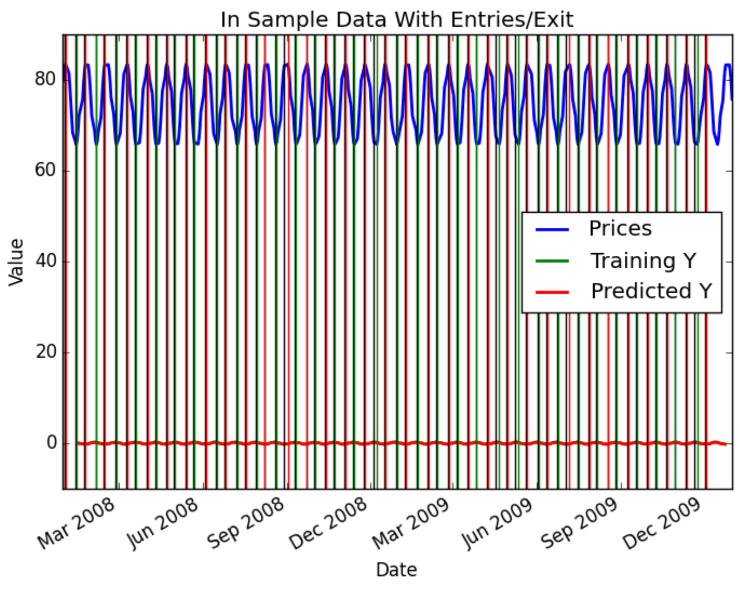

Chart 3 – In Sample Data with Entries/Exit

With the ordering action that we created from the policy and predicted Y, next, we will apply the ordering action back to the market simulator to see how well does it perform comparing to market.

Sine Data In Sample Back test

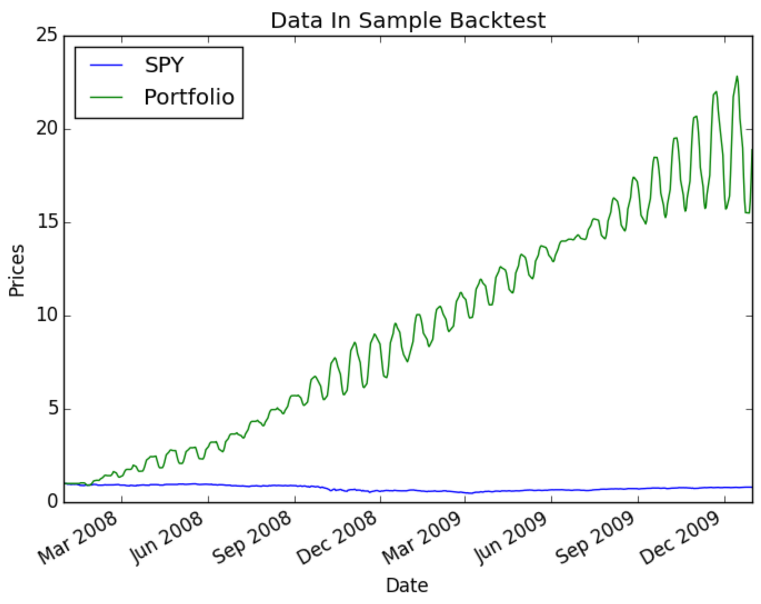

Chart 4 – In Sample Data Back Test

By applying the policy, the back test provided a significant return with the following values:

Sharpe Ratio of Fund: 2.76310271763
Sharpe Ratio of SPY: -0.149575888341

Cumulative Return of Fund: 17.88838836
Cumulative Return of SPY: -0.201395139514

Final Portfolio Value: 188883.8836

ML4T_399 Out Sample Testing Learner

In this part, we will test our learner with year of 2010 to see how well our learner can performs with the new dataset. At first, we will build our ordering action using the prediction values that were generated using our existing learner and the new dataset.

Sine Data Out of Sample Entries/Exits:

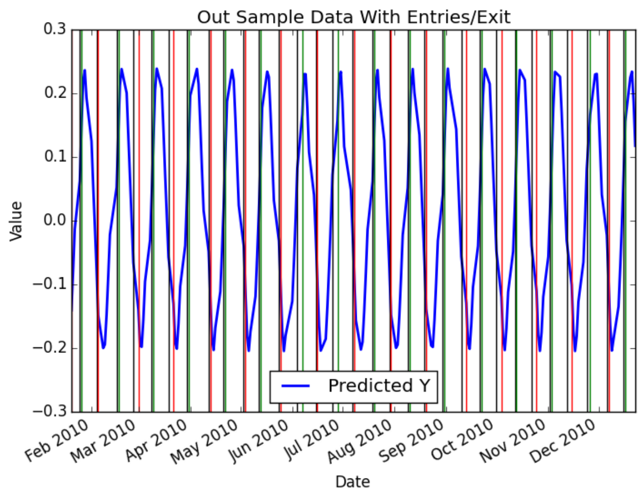

Chart 5 – Out Sample Data with Entries/Exit

Sine Data Out of Sample Back Test

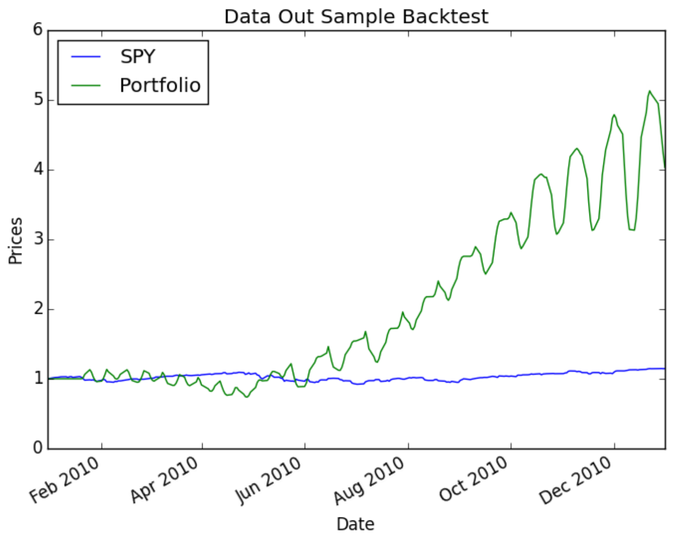

Chart 6 – Out Sample Data Back Test

Although the improvement was not as good as in sample back-test, the out sample back test still provide a significant return with the following values:

Sharpe Ratio of Fund: 2.18072297543
Sharpe Ratio of SPY: 0.848235559397

Cumulative Return of Fund: 3.03179189
Cumulative Return of SPY: 0.145862684324

Final Portfolio Value: 40317.9189

IBM Testing Learner

In this part, we will test our learner with year of 2010 with new stock to see how well our logic can performs with the new dataset. Here are some features that I used to run the training and testing process for IBM stocks:

- Data: **IBM**
- Training Starting Date: **2007-12-31**
- Training Ending Date: **2009-12-31**
- Testing Ending Date: **2009-12-31**
- Testing Ending Date: **2010-12-31**
- Learner: **Linear Regression**
- Technical Indicators: **Bollinger Bands, Momentum, Volatility**
- Window's Size: **12**
- Portfolio's Starting Value: **$10,000**

IBM Data In Sample Entries/Exits:

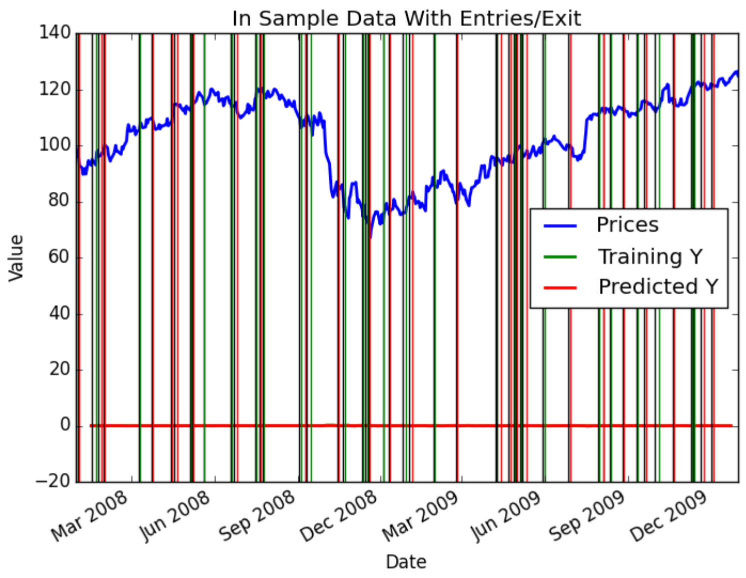

Chart 7 – IBM In Sample Data with Entries/Exit

IBM Data In Sample Back Test

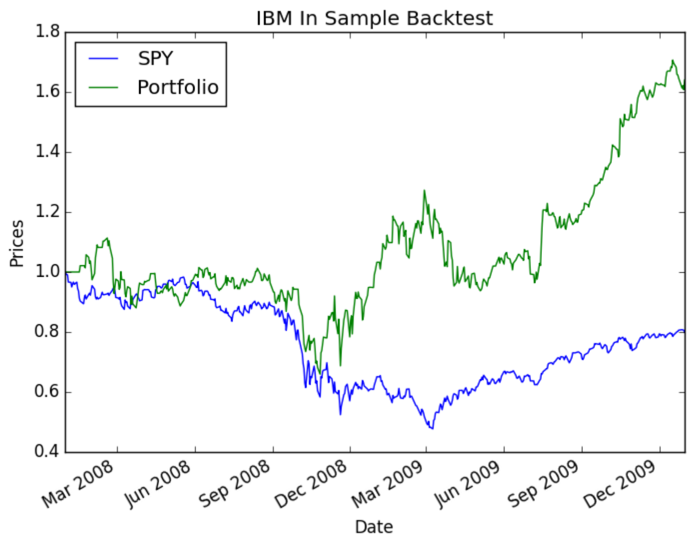

Chart 8 – IBM In Sample Data Back Test

<u>Detailed Result:</u>
Sharpe Ratio of Fund: 0.786315246985
Sharpe Ratio of SPY: -0.149575888341

Cumulative Return of Fund: 0.6396
Cumulative Return of SPY: -0.201395139514

Final Portfolio Value: 16396.0

IBM Data Out of Sample Entries/Exits:

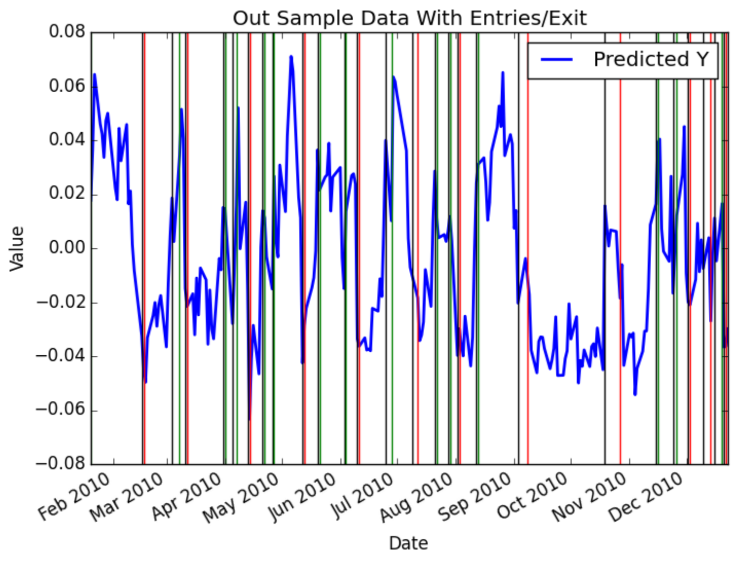

Chart 9 – IBM Out Sample Data with Entries/Exit

IBM Data Out of Sample Back Test:

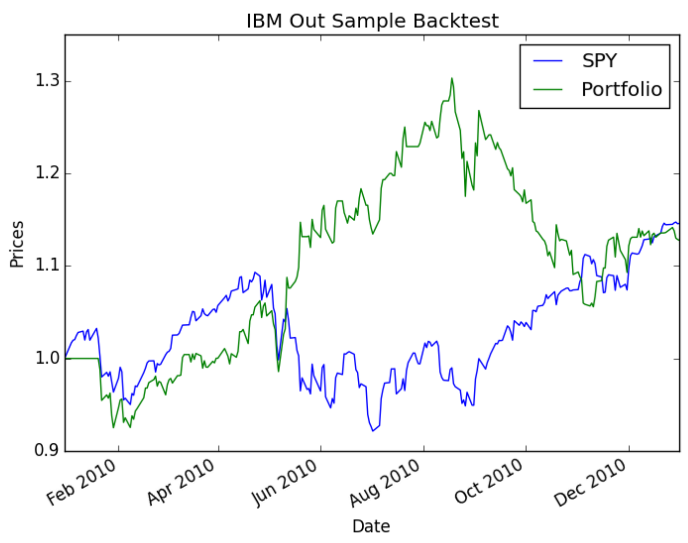

Chart 10 – IBM Out Sample Data Back Test

Detailed Result:

Sharpe Ratio of Fund: 0.681940575365
Sharpe Ratio of SPY: 0.848235559397

Cumulative Return of Fund: 0.1277
Cumulative Return of SPY: 0.145862684324

Final Portfolio Value: 11277.0

Discussion of Results:

Overall, the learner and policy works fairly well with all of testing data. Although the learners works very well with the ML4T_399 dataset with more than 1700% return in sample and 300% out of sample, it only provides 60% return in sample and 12% return out sample with IBM dataset. The reason for the huge difference between those datasets is because the ML4T dataset follows the sign wave and it is more predictable while IBM dataset does not follow any particular patterns. For that reason, the IBM dataset is harder to train and it will return a much lower returns than the ML4T data.

A reason for my learner to works with both IBM and ML4T datasets is the policy. Since in the policy I do not set any limit of how many stocks that can be bought in one position, I decided to make the action on buying and selling stocks when the prediction is good and hold on those stocks for 5 days when prediction is not good. By doing that, I maximize the profit for every opportunity that I can find based on the prediction.

What would you do differently?

If I have more time, I want to calculate the correlation between each feature in the Training data to see whether they are correlated positively or negatively. I will remove any negative correlated features and look for other features that can provide positive correlation.
I also want to try different technical indicators to see if they can provide a better Trainings features for my learner.