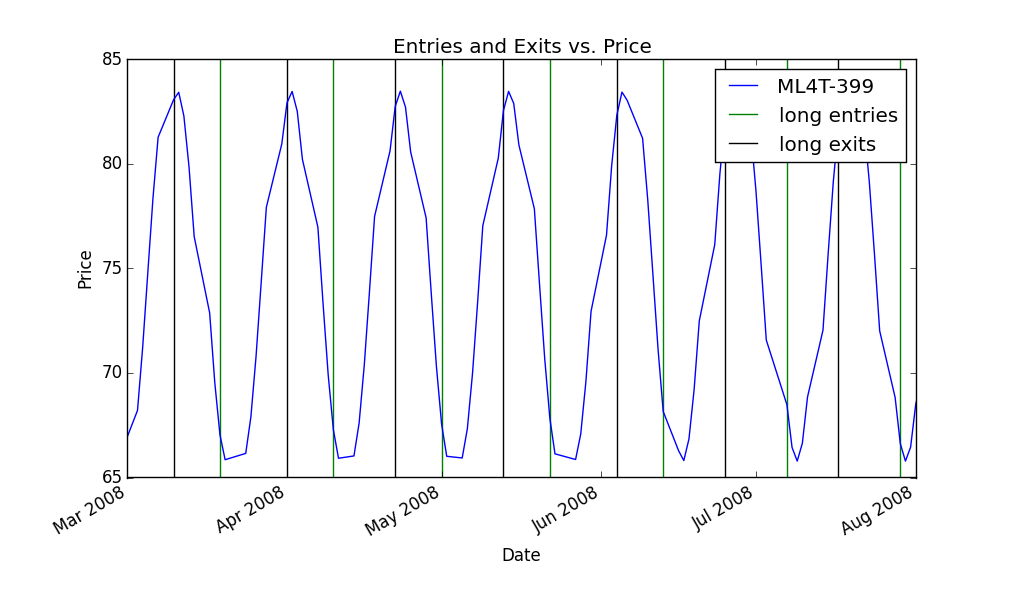
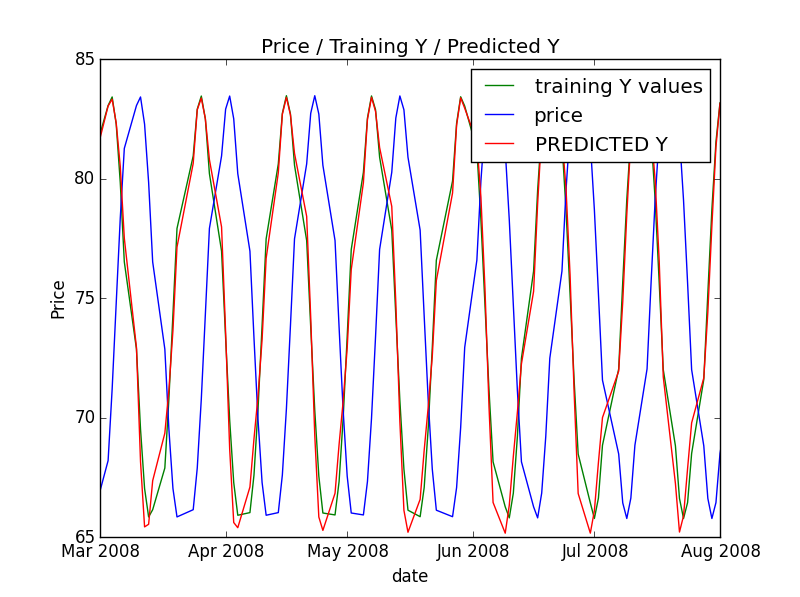
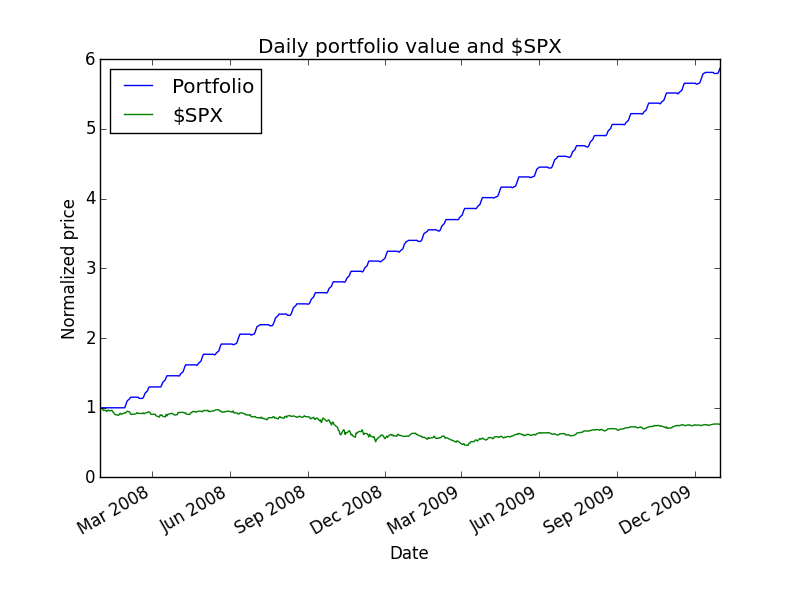
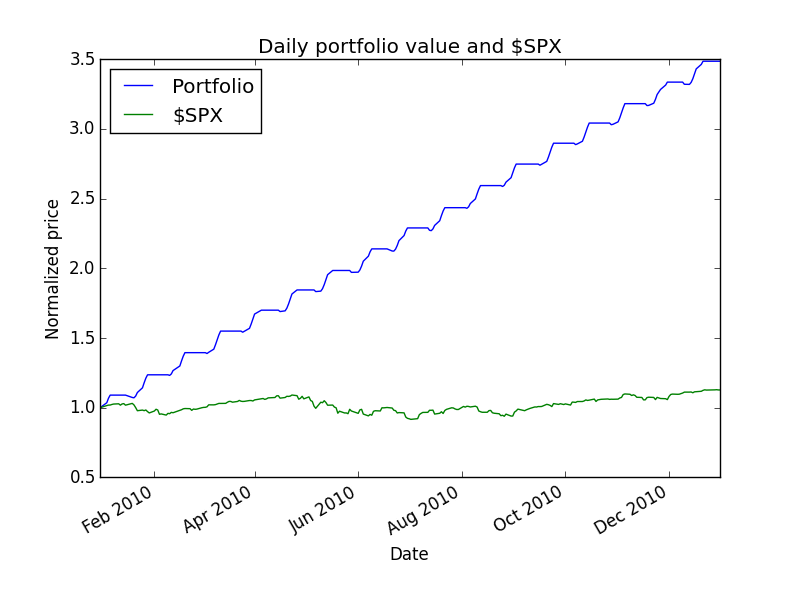
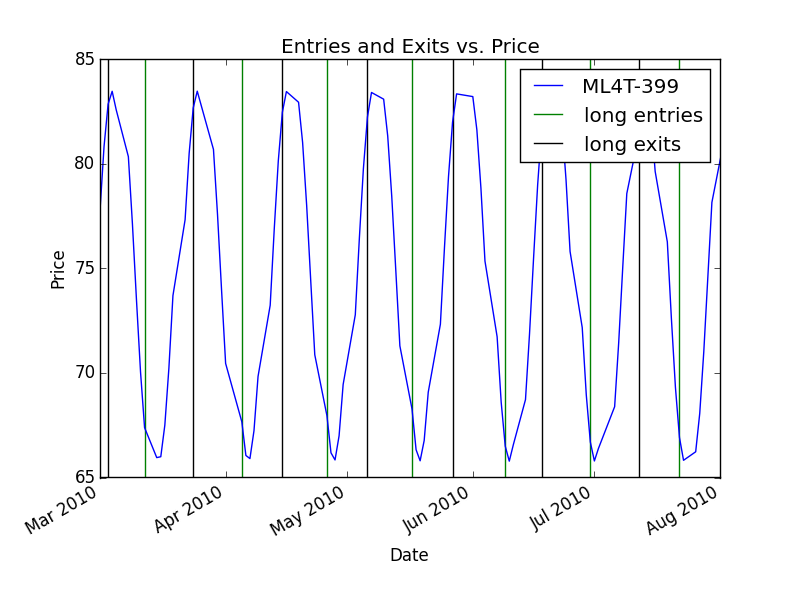
Report.pdf for MC3-2 Daniel Rozen drozen3

**ML4T-399 Charts**

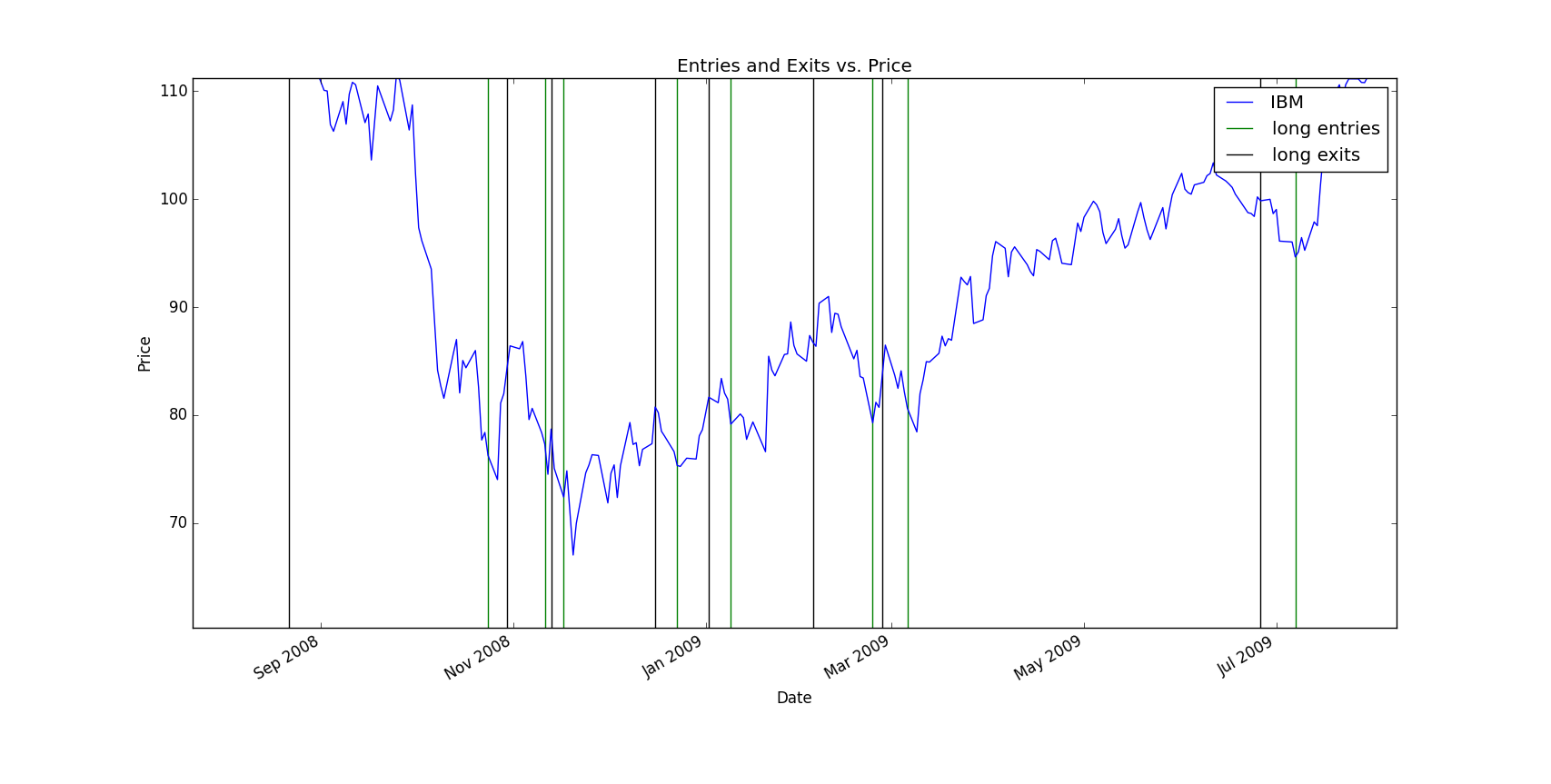




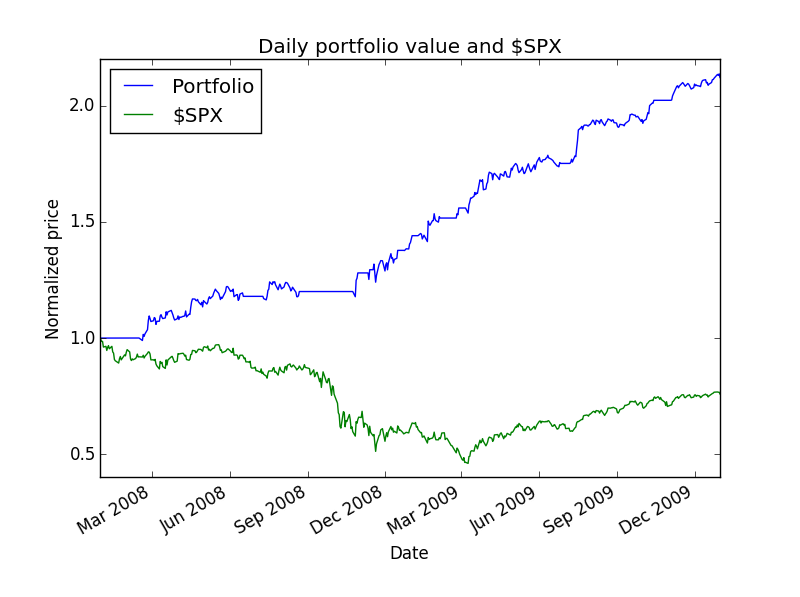
Out of sample:



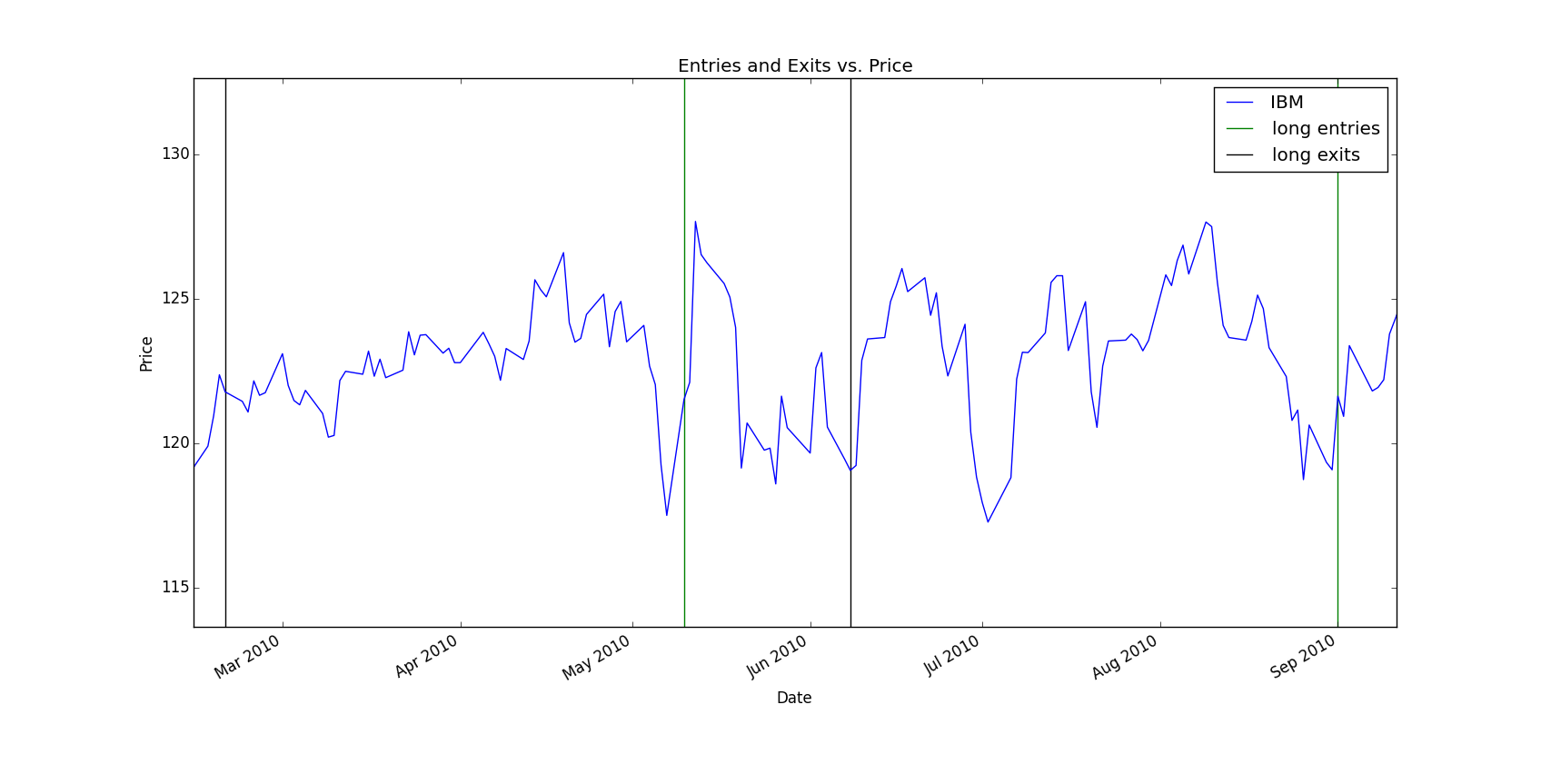
**IBM Charts** 6. IBM IN SAMPLE ENTRIES/EXITS



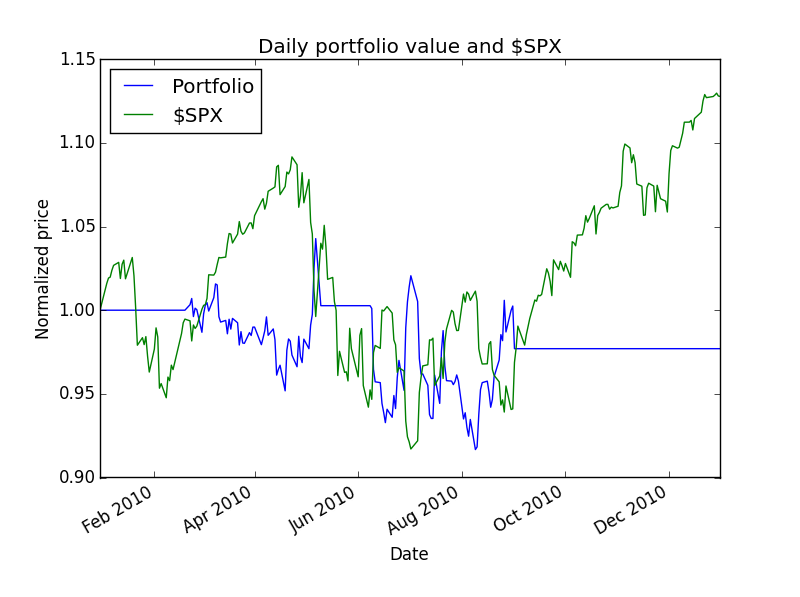
7. IBM IN SAMPLE BACKTEST



8. IBM OUT OF SAMPLE ENTRIES and EXITS



9. IBM OUT OF SAMPLE BACKTEST



**Describe each of the indicators you have selected in enough detail that someone else could reproduce them in code.**

The indicators I have chosen are the folowing:

Bollinger value

*# bb\_value[t] = (price[t] - SMA[t])/(2 \* stdev[t])*

bb\_value = (price - SMA) /(2\*std)

*#momentum[t] = (price[t]/price[t-N]) - 1 N = # days*

momentum

momentum = price/price.shift(N) - 1

volatility

*# volatilty[t] = stddev(dailyreturns)*volatility = pd.rolling\_std(daily\_returns, window=20, min\_periods = 20)

* Describe your trading policy clearly.

When 5 day price increased > 2%, long entry and when decreased < . 2% long exit.

* Discussion of results. Did it work well? Why? What would you do differently?

Worked very well with ML4T-399 data. Almost increased 6 times in portfolio value in sample ,and 3.5 times out of sample. Worked quite well with IBM in sample, almost doubled in portfolio. However, lost $330 from $10,000 for out of sample. Predictions of the KNN learner weren’t very accurate for out of sample, and therefore the long entries and exits weren’t made at the most appropriate times.

The last 70 or so values in the KNN learner became a constant which led to inaccuracy.

“? What would you do differently?”: I would’ve combined linear and KNN learners with bagging to achieve higher accuracy and therefore better results with more appropriate entries and exits.

I also would’ve spent more time researching indicators and trying out different combinations for more optimal results.