Pranshav Thakkar

pthakkar7

# Strategy Learner Report

The goal of this assignment was to frame the trading problem as a problem for a learner of our choice. I decided to use a Random Forest regression learner by implementing a BagLearner with 50 bags of Random Tree learners. Now, in order to train our learner we would have to provide it with an X (list of features) and a Y (outputs we want given X). So, similar to manual strategy, we want to have some indicators based on the data that give us signals on whether to buy, sell, or do nothing, and these indicators would be the features (X) that the learner uses. I used the same two indicators that I used for manual strategy: Price/SMA ratio and Bollinger Band Percentage ( (prices – bottom band)/(top band – bottom band) ). Both of the indicators have lookback windows of 14 days. Now, we have to give our learner Y values to train with based on these features, and I decided that these Y values would be either 1: buy stock, -1: sell stock, and 0: do nothing. In order to figure out which signal to put in Y for a given day, I decided to look at the ratio of the future N day prices over the current prices, as this will give us a general indication on whether the stock price is going up or going down, and it also relates nicely to the indicators that I have chosen. If the ratio was greater than 0.01 (YBUY), then I would put a 1 into Y. If the ratio was lower than 0.01 (YSELL), then I would put a -1 into Y. If it was between these values, I would put a 0 (do nothing). I also took into account market impact for making these classifications, so the ratio would have to be greater than (YBUY + impact) for a buy and less than (YSELL – impact) for a sell. What this accomplishes is that it trains our learner to give us a signal to buy, sell, or do nothing based solely on the indicators, and learn what to do when it sees these indicators again.

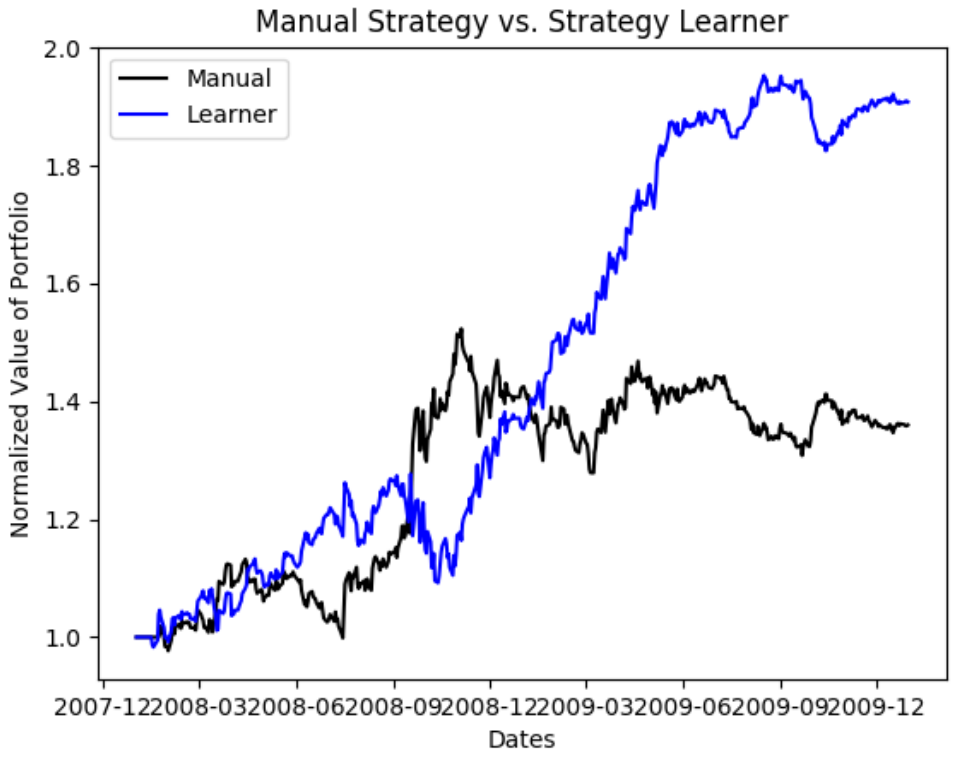
When we use our learner to predict on indicators and data it hasn't seen before, the Random Forest predicts values to give, and our Bag Learner takes the mode of these 50 trees in the forest and gives us our Y results that we can use to make our trades. Now, as the trees are regression learners, we won't get exact 1s, 0s, and -1's all the time in our Y. So, I decided that a Y value of greater than 0.5 indicates a BUY, a Y value of less than -0.5 indicates a SELL, and anything in between indicates a DO NOTHING. Based on how many shares we are holding, we then decide whether to buy or sell 1000 or 2000 shares, as we can have a max net holding of 1000 shares either LONG or SHORT. Because the Random Forest gives us a regression value so easily and we take the mode of all the possible values, we generally end up taking the best action and thus no other manipulation of the data, like discretization or standardization, is needed.

**Experiment 1**

For this experiment, we look at the symbol JPM over the time frame from January 1, 2008 to December 31, 2009. Our portfolio has a starting cash value of $100,000. We assume that commission and market impact are both 0. We then form a set of trades to execute based on our manual strategy from before, and we also train our strategy learner over the same time frame, and get a set of trades to execute from our learner. We plug both of these trade orders into our market simulator separately, again with the parameters of commission and market impact set to 0. We get the values for the portfolios from the simulator, and we normalize both of them with respect to the first day of the time frame, so that we can compare them easily.

As this time frame is considered an in sample (where our learner trains and tests on the same data), we can expect that it will have a very good outcome, better than that of the manual strategy. Indeed, this is what we see, as our learner has a few ups and downs about halfway through the time period but eventually shoots up ahead of the manual strategy. Our learner ends up with about 1.9 times the value of the portfolio we started out with, while the manual strategy ends up with about 1.4 times the value of the starting portfolio.

We can expect this same relative result almost all the time with an in sample time frame, as a machine learning learner will always do well on data it has seen before and trained on. In fact, if we train the learner on the same data repeatedly, it can quite often converge to an optimal strategy, and quickly too.



**Experiment 2**

As market impact goes up, we generally become less willing to make certain trades, as our profit becomes smaller. So, I believe that as market impact increases, our learner will be more resistant to making trades and end up with a lower portfolio value.

To conduct the experiment, I created 6 different learners that accounted for impact values from 0.0 to 0.1 in steps of 0.2 respectively, and then trained and tested each of them on the in sample time frame of January 1, 2008 to December 31, 2009 on the JPM symbol. As far as results go, we already saw our strategy learner earn 1.9 times the portfolio value with impact 0 in experiment 1, and like I predicted, the portfolio values for the learners with higher impacts is less than that one. However, the values are much, much lower than I expected, with the 0.08 impact learner ending up with almost -1 times our starting value, which means it's lost almost all of our money. I believe this result happened because I account for impact in the learner in a way where it makes the learner more conservative in making trades, as it waits for the future n day market return to be either better than YBUY + impact to buy or lower than YSELL – impact to sell. Because of this, our learners don't maximize the value by making certain trades, instead choosing to hold more often. Sometimes this conservative approach works decently okay, as with impact 0.1, but other times it can backfire drastically when you don't hit the right trades, as with impact 0.08.