Machine Learning for Trading: Strategy Learner Report

Sagarika Srishti: ssrishti3

December 2, 2019

1 Framing the trading problem as a learning problem

To accomplish this task, I am taking the help of the Random Forest learner that we had implemented in the Project: Assess Learners. I use bagging to train a forest of 20 random trees, with leaf size as 5, using the Bag Learner, also implemented as part of that project.

For this project, we have to make a learner, that given an input of a stock symbol and the trading date range, outputs a trades dataframe which contains the trades that should be made on each day in order to maximize the cumulative return. To make this a learning problem, as suggested in the project description, we can train the Random Forest learner, in which the X_train values will be the stock price and the indicator values for each day in the in-sample period, and the Y_train values will be the +1, 0, or -1, based on whether the decision for that day should be to go Long, no trade, or to go Short, respectively, as per the n-day returns. I have used 10-day returns and the impact value to determine my trading decisions at training. Once the 10-day returns are calculated, the trading signals are generated as:
if return[i] > impact: go Long
if return[i] < impact: go Short
else: no trade

We will choose the leaf size as 5 to prevent over-fitting of the random forest. Once the learner is trained in this way, we will similarly pass the X_test values to the learner as the stock price and calculated indicators for the out-sample period, and the learner will then predict the trades. Also, since our RTLearner was implemented as a regression learner, we will use the mode of values output by the various random trees to be the predicted class. Once we obtain Y using the strategy learner, we determine the trades based on the Y value for each day, taking care that the current holdings remain -1000, 0, or 1000.

The indicators used to train my learner are SMA, momentum and Commodity Channel Index, all with a window of 20. I did not need to make any adjustments to the data other than normalizing the stock prices. This decision stemmed from the choice of my learner. Since random trees use the median feature values to make split decisions, we do not need to scale or discretize the data.

2 Experiment 1

In this experiment, we compare the performance of the Strategy Learner with the Manual Strategy that we had devised using indicator values in a previous project. I have used the impact value of 0.005 (0.0 for benchmark), starting cash of $100,000.0, commission of $0.0 and compared the performances of these two learners with the benchmark for the stock symbol 'JPM', for the period January 1, 2008 to December 31, 2009. All other settings for the Strategy Learner remain the same as defined in the first section. As we can see in the below graph, Strategy Learner outperforms Manual Rule-based Strategy in the in-sample period. Also given are in-sample statistics for all the three cases.

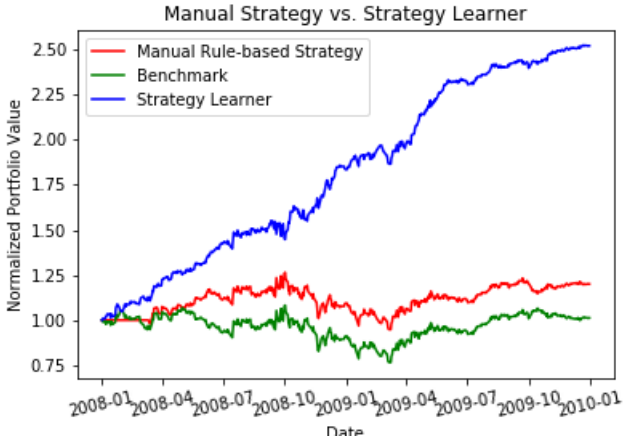

Figure 1: Manual Strategy vs. Strategy Learner for in-sample

Measure	Strategy Learner	Manual Strategy	Benchmark
Cumulative Returns	1.450025	0.200308	0.0123
Std. Dev. of Daily Returns	0.009709	0.01397	0.017004
Mean of Daily Returns	0.001826	0.000459	0.000168

Strategy Learner performs quite better than Manual Strategy and benchmark in the in-sample period, and I think it will always perform better in the in-sample

period, which I saw while experimenting with different date ranges. For the out-sample period, the results might be different as the Strategy Learner might be overfitting in in-sample data. It is also possible for the Strategy Learner to even perform worse than the benchmark in the out-sample period, as seen in the below graph.

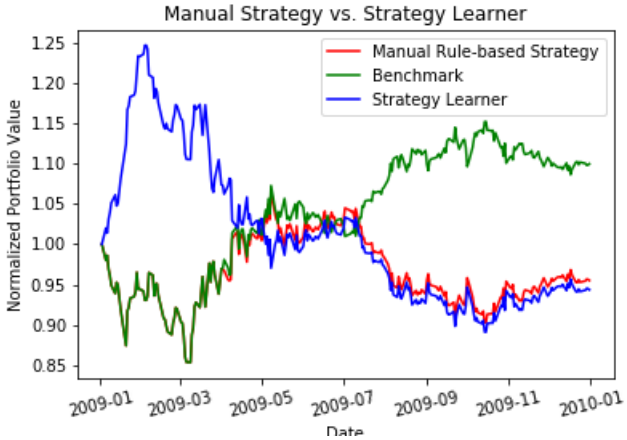

Figure 2: Manual Strategy vs. Strategy Learner for out-sample

3 Experiment 2

In this experiment, we check how the in-sample behavior of the strategy learner changes with impact values. Since impact is a cost that a trader has to bear for every trade they make, I think that a low impact should be beneficial to the trader, i.e., the performance of the strategy learner should be inversely proportional to the impact values. We will use cumulative return and Sharpe ratio to measure this.

To test this, I tested Strategy Learner for an in-sample period of January 1, 2008 to December 31, 2009 for the stock symbol 'JPM', with a starting cash of $100000.0. The commission is set to zero, and we test the learner with impact values of 0.05, 0.005, and 0.0005. All other settings for the Strategy Learner remain the same as defined in the first section. We plot the performance of these learners along with the defined benchmark for comparison. The graph and statistics given below corroborate our hypothesis. The Strategy Learner's performance increases with decrease in the impact value. For the benchmark, impact is taken to be 0.0. The portfolio returns are maximized when impact is the minimum, that is, 0.0005, and they decrease as the impact is increased.

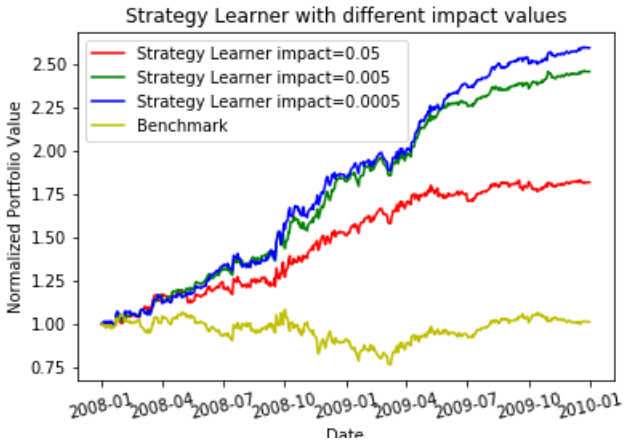

Figure 3: Strategy Learner for different impact values for in-sample

Learner	**Impact**	**Cumulative Returns**	**Sharpe Ratio**
Strategy Learner	0.05	0.50782	1.043778
Strategy Learner	0.005	1.548855	3.390599
Strategy Learner	0.0005	1.732578	3.290479
Benchmark	0.0	0.0123	0.157074