Wen Cheng Chai

wchai8

Wchai8@gatech.edu

Strategy Evaluation Project

Introduction

For project 8, an intuition-based manual trading strategy was created utilizing the Bollinger Band Percentage, Momentum, and Price/SMA ratio as indicators. The indicators combined with a set of rules on certain values generated long or short signals. Another trading strategy was devised using an ensemble learner combination of a Random Tree Learner and bagging method, which is labeled as our StrategyLearner.py These trading strategies were then tested on two separate experiments. Before conducting the tests to see which strategy performs better, I will make a couple of hypotheses. We will hypothesize that the manual trading strategy will perform better than the benchmark strategy for in-sample and out of sample data. For experiment 1, I hypothesize that that strategy learner will outperform the manual strategy. For experiment 2, I hypothesize that that strategy earner will perform better when there are lower levels of impact. The reasoning for these hypotheses will be addressed more in the Experiment section.

Indicator Overview

The 3 indicators that were chosen to build both trading strategies are the Bollinger Band Percentage(BB %), momentum indicator, and Price/SMA ratio. They will be further discussed in this passage.

1. Volatility

Volatility can be considered a price range where a stock moves over a certain period of time. It can also be a measure of risk or uncertainty about a stock. When volatility is high, it can be determined that investing in this stock is risky. However, this also means there is a higher possible return for the investor but the risk is significantly higher. In our strategy, I aim to take advantage of buying a stock when volatility is low, and selling it when volatility is high. To optimize this, we need to optimize the parameter of daily returns, and essentially buy in price ranges when the daily returns are not changing significantly. Volatility is measured as the standard deviation of prices which is represented in equation 1 below:

$$Volatility = \sigma = \sqrt{Daily\ Returns} \text{ [Equation 1]}$$

2. Bollinger Bands Percentage

The Bollinger Band Percentage is a metric that quantifies a stock's relative price to the Bollinger bands. According to a reference from barchart.com[1], there are 6 basic relationship levels of interest:

1. %B equals 1 when price is at the upper band
2. %B equals 0 when price is at the lower band

3. %B is above 1 when price is above the upper band
4. %B is below 0 when price is below the lower band
5. %B is above .50 when price is above the middle band (20-day SMA)
6. %B is below .50 when price is below the middle band (20-day SMA)

To maximize the Bollinger Bands, we need to pay attention to the moving window parameter. This is because the Bollinger bands use the SMA to calculate the upper and lower band. This means the SMA has to be accurate for the Bollinger Bands to be an effective tool, so the moving window parameter must be fine tuned.

The equation to calculate the Bollinger Bands Percentage is found in equation 2 below:

$$BBPercentage = \frac{prices - SMA}{2 * \sigma(prices)} \text{ [Equation 2]}$$

3. Price/SMA ratio

The Price/SMA is calculated via the simple moving average. The simple moving average is a leading technical indicator which indicates that it uses historical price data to predict future price movement. The SMA is a rolling mean of stock prices over a period of time which is indicated in equation 3 below.

$$SMA^{(n)}{}_t = \frac{1}{n}\sum_{i=0}^{n-1} Price_{t-i} \quad \text{[Equation 3]}$$

In terms of parameters used to optimize the SMA indicator, the moving window parameter must be adjusted. For example, some stocks may require a lookback period of 10 days instead of 21 days that track the price of a stock better.

The Price/SMA is thus calculated by dividing the prices by the SMA price by time period. The Price/SMA ratio can be useful, as it can mark buy and long signals depending on the ratio.

Manual Strategy(Indicator Choice Explanation and Strategy)

1. Price/Sma Ratio

As mentioned earlier, the price/sma is considered a leading technical indicator which uses historical date to predict future price movement. When the price/SMA ratio is >1, this means that the current price is higher than the average price over an analyzed time period which could imply it is overbought and should be sold. When the price/SMA ratio is <1.0, this means that the stock is below the average price which could mean it is oversold. For my manual trading strategy, the following thresholds for Price/SMA ratio are below:

When Price/SMA Ratio > 0.7, this will generate a buy/long signal.

When Price/SMA Ratio > 1.1, this will generate a sell/short signal.

2. Bollinger Band Percentage

With reference to the Bollinger Band Perentage explanation above, the Bollinger Band Percentage is a metric that determines a stock's relative price to the Bollinger Bands. The range of the metric is from 0 to 1, where 0 being the lower band and 1 being the upper band. When the price is closer to the upper band(BBP% = 1) this can mean the stock is overbought. Conversely, when the price is closer to the lower band(BBP% = 0) this can mean the stock is oversold. For my manual trading strategy, the following thresholds for the Bollinger Band Percentage are below:

When BBP % > 0.8, a short/sell signal will be generated

When BBP % <0.2, a buy/long signal will be generated.

Volatility

As mentioned earlier, volatility can be used as a metric for risk when owning the stock. For my manual strategy, I want to buy when the volatility is low as there is lower risk and sell when volatility is high when it becomes too risky to hold. The thresholds for volatility for the manual trading strategy are below:

When volatility < 0.025 this generates a buy/long signal

When volatility > 0.075 this generates a sell/short signal.

Manual Strategy Method

To summarize above, the indicators were combined to generate a manual trading strategy. The manual trading strategy generates buy signals for the following indicators:

When BBP % > 0.2, Price/SMA ratio > 0.7, and volatility < 0.025 this generates a buy/long signal

When BBP % <0.8, Price/SMA Ratio < 1.1, and volatility <0.075 this generates a sell/short signal.

Whenever neither of these indicators generates a signal, the strategy is to then hold the stock until a signal is generated.

It is also important to note that this strategy also has a holdings threshold that cannot exceed 2000 or -2000 shares.

Plots of In-Sample and Out-of-Sample Results Manual Strategy Vs Benchmark

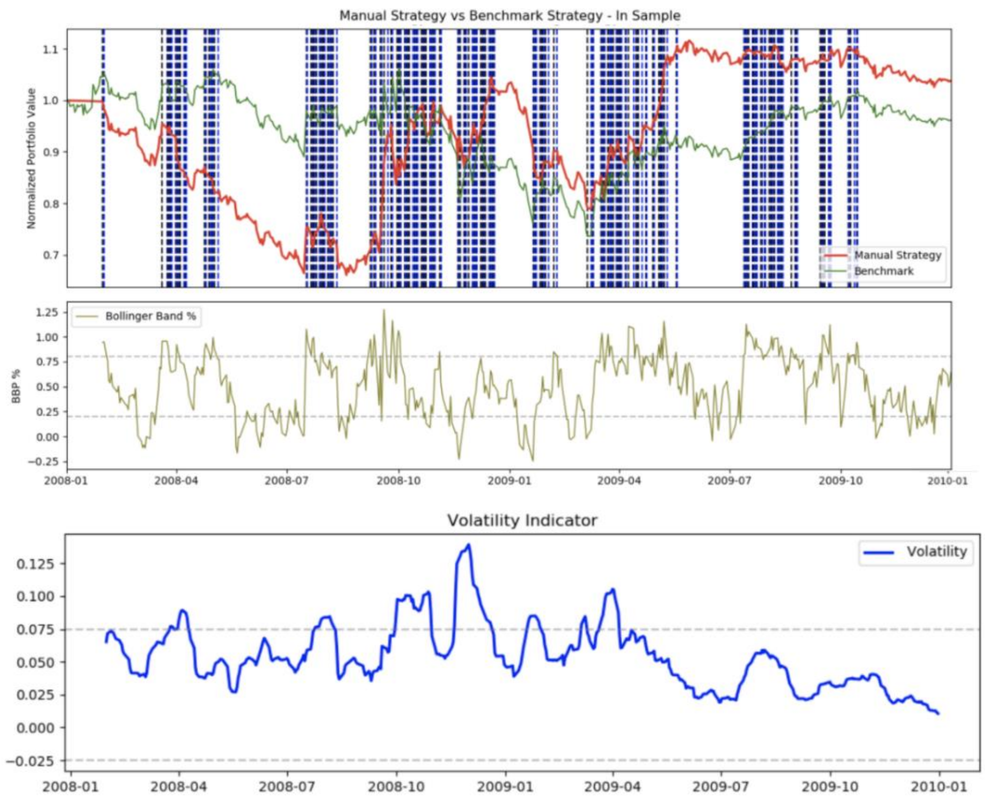

Figure 1: Manual vs Benchmark Strategy

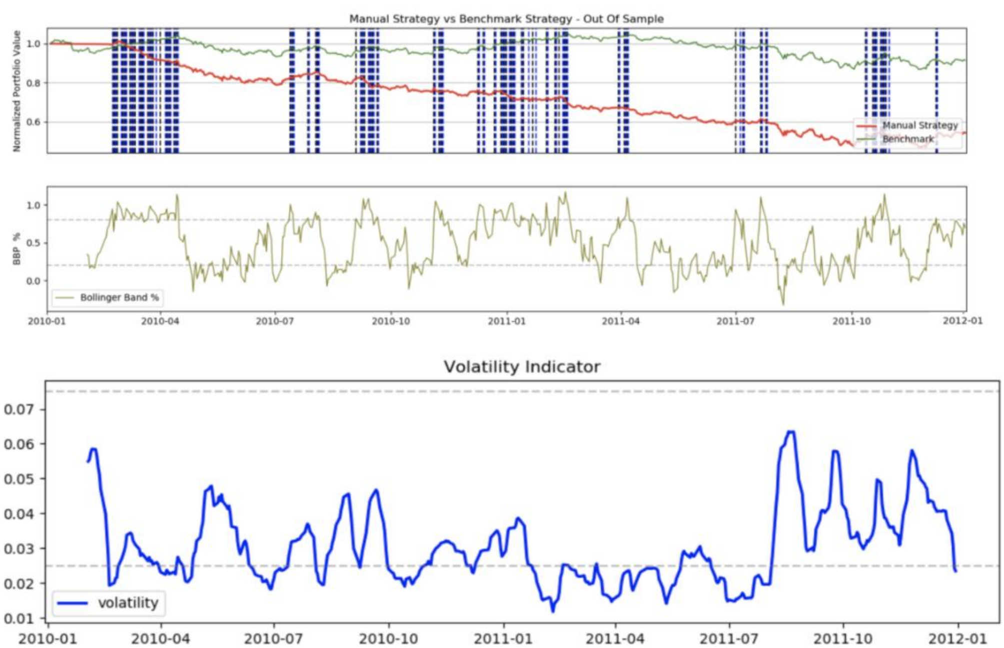

Figure 2: Manual vs Benchmark Strategy Out of Sample

Note: All trades are indicated by black and blue lines. Black = Short, Blue = Long

Performance Table for Manual Strategy and Benchmark Strategy for In-Sample Data

Statistics	Manual Strategy	Benchmark
Cumulative Return	**0.038508**	**-0.037924**
Standard Deviation of Daily Returns	**0.017607**	**0.017468**
Mean of Daily Returns	**0.000228**	**0.000075078**

Performance Table for Manual Strategy and Benchmark Strategy for Out of-Sample Data

Statistics	Manual Strategy	Benchmark
Cumulative Return	**-0.456169**	**-0.085308**
Standard Deviation of Daily Returns	**0.012402**	**0.008501**
Mean of Daily Returns	**-0.001133**	**0.-0.00014**

Looking at the table above, we can find metrics such as cumulative return, standard deviation of daily return, and mean of daily returns. These metrics help us evaluate things such as the overall perforance of the portfolio, risk of the portfolio as well as how much return is expected with respect to that risk. When looking at the data, you can see that the Manual Strategy is superior to the Benchmark data in all metrics for the in-sample data. This is because when creating the manual strategy, we had trained and tested our Manual Strategy with the same data set. However, when we start to observe how the manual strategy performs in out-of-sample data, we notice that the manual strategy performs worse than the original buy and hold strategy. This is a potential outcome, as our model has not been trained with any out of sample data. To further improve this strategy's performance on out of sample data, the first step we could take could simply be using a cross-validation method in order to build our model. Another reason why our model performs better in-sample is because of selection bias. In this case, selection bias may exist in the sample testing phase, as I attempted to select the parameters that gave the best performance. Another potential reason why there is a discrepancy is because of data snooping bias(future peeking bias in Professor Balch's lectures). This could happen particularly when we backtest on data to find a strategy that performs better than the benchmark.

Strategy Learner

For the strategy learner, I used an ensemble learner that utilized a Random Tree Learner to perform the learning. The Bag Size for the bag learner will be 20 with a leaf size of 5. Thus, the trading problem has been converted into a learner problem.

Data Standardization/Discretization

To standardize the data, the mode of the results is taken from the ensemble of random trees for our strategy learner implementation. There is no need to discretize the data as I am not using a Qlearner.

Hyper Parameters

For our Strategy Learner, the most important hyper parameter is the number of decision trees used in the ensemble's Random Forest learner. In general, more trees means better results, but it comes at the cost of time to generate all these trees and to sort through the data. Therefore, it is important to select a small or "minimal leaf size". For the bag learner, the most important hyper parameter is the number of bags. For the random forest learner, I came up with the idea of using a leaf size of 5. This is because according to rapidminer documentation(a data science platform) a good minimal leaf size for regression based problems is 5 [2]. As for the bag size, a bag size of 20 was chosen as mentioned earlier. To determine this size, I tested the strategy learner on the same data with different bag sizes to see which number of bags produces the best results.

Training and Testing

The Random Tree learner will generate multiple random decision trees based on mappings of indicator values to the chosen indicators. The following functions for training and testing in the code and their parameters will be discussed in this passage step by step.

Training Phase(Adding data to add_evidence() method):

For the training phase, use the add_evidence() method. The parameters for this method are below:

Stock symbol, start Date, end Date, start Value.

Steps:

1. Calculate the indicators selected, price/SMA ratio, BBP %, and volatility.

2. Create an xTrain dataset by combining all the indicator values into a dataframe.

3. Compare the N-Day change of price with a market variance of 2%. Depending on the comparison, this will add a 1,0, or -1 that represents 3 states. These 3 states are buy, hold, or sell. This will represent our yTraining data.

Testing Phase(Utilizing the testpolicy() method)

For the testing phase, the testpolicy() method is used. The parameters for this method are below:

Stock Symbol, startDate, endDate, startValue

Steps:

1. Calculate the technical indicators

2. Create an xTest data set using the values from the calculated technical indicators

3. Generate yTest values from calling the query() method from the ensemble learner

4. The dataset returned willl be a list of trades that occur, which will be the constraints between a holdings value of -2000 to 2000.

Experiment 1 (Manual/Strategy Learner)

For experiment 1, I am required to compare the manual strategy trader's performance, the benchmark performance, and the new ensemble learner strategy. Both of these strategies utilize the same indicators and thresholds to create a case for a consistent comparison. The stock that was analyzed is JPM stock from the date ranges of January 2008 to December 2009. As for our parameter values, the starting value for our portfolio is $100,000 and the impact of trades on the stock price is about 0.005. The moving window is 21 and the comission is 9.95. For our hypothesis, we will hypothesize that the strategy learner(Bag ensemble) will outperform the Manual Strategy we have created. As for assumptions, we will assume we can create our analysis based on our in-sample data.

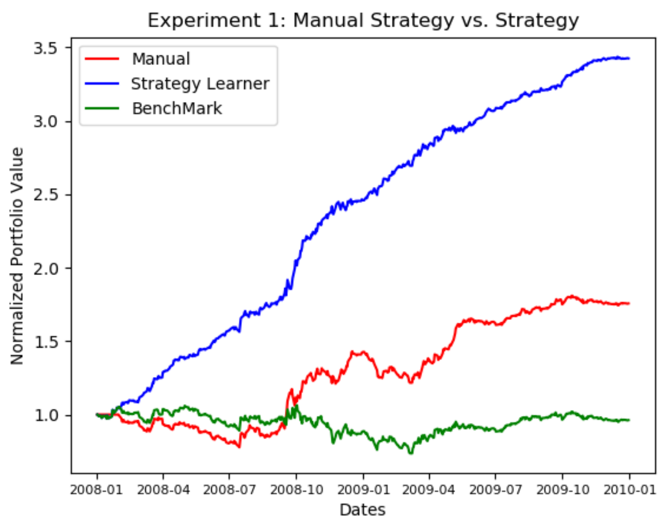

Figure 2: Experiment 1

From the data above, the normalized portfolio value performance is plotted for JPM stock utilizing the Manual Strategy, Ensemble Strategy(Strategy Learner), and Benchmark strategy. We can see that our hypothesis aligns with the results in the chart above. Looking at the chart, we can see that the strategy learner performs the best out of the three strategies. The benchmark strategy performs the worst, while the Manual Strategy performs in the middle. However, it is important to note that the overall performance of the Strategy Learner is significantly higher than the Manual strategy(almost double).

In terms of expectations for the outcome for in-sample data, I would expect that the Strategy Learner will consistently beat the Manual Strategy. This is due to the fact that the performance of the Strategy Learner is due to randomization, while the benchmark and Manual Learner are not. The randomness helps in the process of choosing the best factors to split for the random tree, which makes sense why its performance outclasses the other strategies. In theory, random forest learners can also generate bad trees but the probability of this happening is extremely low.

Experiment 2

In this last experiment, the impact value's effect on in-sample trading behaviour is analyzed. My hypothesis is that the trader's performance will be highest when the impact is low. As impact is a measure of how much the price moves against a trader when referencing the historical price data for all transactions. We will be testing the impact solely on our Strategy learner. For our strategy learner, we will use a leaf size of 5 and a bag size of 20. The moving window is 21, the stock analyzed will be JPM, and the time frame is from January 2008 to December 2009. The impacts we will be analyzing are: 0.0, 0.001, 0.01, and 0.1. The results are shown in the figure below for the performance of our strategy learner with respect to different impact

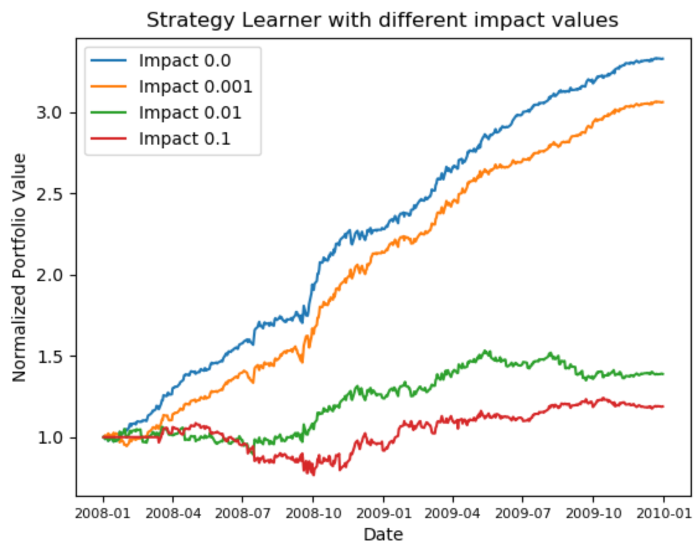

Figure 3: Experiment 2

From the plot above, you can observe that the strategy learner indeed performs better when impact is lower. When the impact is lowest(0.001), you can see that the return of the portfolio is almost at the same level as if there was no impact, indicating that lower impact is indeed better for the trader. It is also interesting to observe how sensitive the learner is with respect to impact. When looking at the results from impact 0.001 to 0.01(magnitude of 10 increase), the normalized portfolio value is drastically different. However, going from impact 0.01 to 0(another magnitude of 10 increase), we notice the normalized portfolio value does not change much in value. In the table below, you can also observe further how impact affects other portfolio metrics such as Cumulative Return, Volatility, and the Sharpe Ratio.

Impact	**Cumulative Return**	**Standard Deviation**	**Sharpe Ratio**
0.0	2.3303	0.0083419	4.6135
0.001	2.0630	0.0093186	3.8613
0.01	0.3889	0.0145759	0.8253
0.1	0.1886	0.0182783	0.4437

When looking at the table of metrics with respect to impact, we find the results we expected. The Sharpe Ratio decreases as the impact increases. The cumulative return also decreases as impact increases. Moreover, the standard deviation of prices(volatility) increases when impact increases. From a risk and profit perspective, trading with as little impact as possible is ideal.

References:

https://www.barchart.com/education/technical-indicators/bollinger_bands_percent[1]

GmbH, RapidMiner. "Random Forest (Concurrency)." *Random Forest - RapidMiner Documentation*, https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/trees/parallel_random_forest.html. [2]