Fall 2019 Project 6: Manual Strategy

From Quantitative Analysis Software Courses

Contents

Revisions

This assignment is subject to change up until 3 weeks prior to the due date. We do not anticipate changes; any changes will be logged in this section.

Overview

In this project you will develop a trading strategy using your intuition and Technical Analysis, and test it against a stock using your market simulator. In a later project, you will use your same indicators but with Machine Learning (instead of your intuition) to create a trading strategy. We hope Machine Learning will do better than your intuition, but who knows?

Template

There is no distributed template for this project. You should create a directory for your code in ml4t/manual_strategy and make a copy of util.py there. You will have access to the data in the ML4T/Data directory but you should use ONLY the API functions in util.py to read it.

You should create the following code files for submission. They should comprise ALL code from you that is necessary to run your evaluations.

- `indicators.py` Your code that implements your indicators as functions that operate on dataframes. The "main" code in indicators.py should generate the charts that illustrate your indicators in the report.

- `marketsimcode.py` An improved version of your marketsim code that accepts a "trades" data frame (instead of a file). More info on the trades data frame below. It is OK not to submit this file **if** you have subsumed its functionality into one of your other required code files.
- `ManualStrategy.py` Code implementing a ManualStrategy object (your manual strategy). It should implement testPolicy() which returns a trades data frame (see below). The main part of this code should call marketsimcode as necessary to generate the plots used in the report.
- `TheoreticallyOptimalStrategy.py` Code implementing a TheoreticallyOptimalStrategy object (details below). It should implement testPolicy() which returns a trades data frame (see below). The main part of this code should call marketsimcode as necessary to generate the plots used in the report.

Note that we may not test your code, so we may not know if you didn't organize your code as recommended, but this arrangement will be required for later projects, so it is worthwhile getting it set up this way. The key requirement is that, if necessary, a TA should be able to run your code on a buffet machine and get the same results (e.g., statistics and charts) that we see in your report.

Data Details, Dates and Rules

- Use only the data provided for this course. You are not allowed to import external data.
- Please add in an author function to each file.
- For your report, trade only the symbol JPM. This will enable us to more easily compare results.
- You may use data from other symbols (such as SPY) to inform your strategy.
- The in sample/development period is January 1, 2008 to December 31 2009.
- The out of sample/testing period is January 1, 2010 to December 31 2011.
- Starting cash is $100,000.
- Allowable positions are: 1000 shares long, 1000 shares short, 0 shares.
- Benchmark: The performance of a portfolio starting with $100,000 cash, investing in 1000 shares of JPM and holding that position.
- There is no limit on leverage.
- Transaction costs for ManualStrategy: Commission: $9.95, Impact: 0.005.
- Transaction costs for TheoreticallyOptimalStrategy: Commission: $0.00, Impact: 0.00.
- Correct trades df format used.

Part 1: Technical Indicators (20 points)

Develop and describe at least 3 and at most 5 technical indicators. You may find our lecture on time series processing to be helpful. For each indicator you should create a single, compelling chart that illustrates the indicator.

As an example, you might create a chart that shows the price history of the stock, along with "helper data" (such as upper and lower bollinger bands) and the value of the indicator itself. Another example: If you were using price/SMA as an indicator you would want to create a chart with 3 lines: Price, SMA, Price/SMA. In order to facilitate visualization of the indicator you might normalize the data to 1.0 at the start of the date range (i.e. divide price[t] by price[0]).

Your report description of each indicator should enable someone to reproduce it just by reading the description. We want a written detailed description here, not code, however, it is OK to augment your written description with a **pseudocode** figure. Do NOT copy/paste code parts here as a description.

At least one of the indicators you use should be completely different from the ones presented in our lectures. (i.e. something other than SMA, Bollinger Bands, RSI).

N.B. Be careful of which indicators you end up with. We will require you to reuse the same ones on a future assignment.

Part 2: Theoretically Optimal Strategy (20 points)

Assume that you can see the future, but that you are constrained by the portfolio size and order limits as specified above. Create a set of trades that represents the best a strategy could possibly do during the in sample period. The reason we're having you do this is so that you will have an idea of an upper bound on performance.

The intent is for you to use adjusted close prices with the market simulator that you wrote earlier in the course. For this activity, use $0.00, and 0.0 for commissions and impact respectively.

Provide a chart that reports:

- Benchmark (see definition above) normalized to 1.0 at the start: Green line
- Value of the theoretically optimal portfolio (normalized to 1.0 at the start): Red line

You should also report in text:

- Cumulative return of the benchmark and portfolio
- Stdev of daily returns of benchmark and portfolio
- Mean of daily returns of benchmark and portfolio

Your code should implement testPolicy() as follows:

```
df_trades = ms.testPolicy(symbol = "AAPL", sd=dt.datetime(2010, 1, 1), ed=dt.datetime(2011,12,31), sv = 100000)
```

The input parameters are:

- symbol: the stock symbol to act on
- sd: A datetime object that represents the start date
- ed: A datetime object that represents the end date
- sv: Start value of the portfolio

The output result is:

- df_trades: A data frame whose values represent trades for each day. Legal values are +1000.0 indicating a BUY of 1000 shares, -1000.0 indicating a SELL of 1000 shares, and 0.0 indicating NOTHING. Values of +2000 and -2000 for trades are also legal so long as net holdings are constrained to -1000, 0, and 1000.

Part 3: Manual Rule-Based Trader (50 points)

In `ManualStrategy.py` implement a set of rules using the indicators you created in Part 1 above. Devise some simple logic using your indicators to enter and exit positions in the stock.

A recommended approach is to create a single logical expression that yields a -1, 0, or 1, corresponding to a "short," "out" or "long" position. Example usage this signal: If you are out of the stock, then a 1 would signal a BUY 1000 order. If you are long, a -1 would signal a SELL 2000 order. You don't have to follow this advice though, so long as you follow the trading rules outlined above.

For the report we want a written description, not code, however, it is OK to augment your written description with a pseudocode figure.

You should tweak your rules as best you can to get the best performance possible during the in sample period (do not peek at out of sample performance). Use your rule-based strategy to generate an trades dataframe over the in sample period, then run that dataframe through your market simulator to create a chart that includes the following components over the in sample period:

- Benchmark (see definition above) normalized to 1.0 at the start: Green line
- Value of the rule-based portfolio (normalized to 1.0 at the start): Red line
- Vertical blue lines indicating LONG entry points.
- Vertical black lines indicating SHORT entry points.

We expect that your rule-based strategy should outperform the benchmark over the in sample period.

Your code should implement the same API as above for theoretically optimal:

```
df_trades = ms.testPolicy(symbol = "AAPL", sd=dt.datetime(2010, 1, 1), ed=dt.datetime(2011,12,31), sv = 100000)
```

Part 4: Comparative Analysis (10 points)

Evaluate the performance of your strategy in the out of sample period. Note that you **should not** train or tweak your approach on this data. You should use the classification learned using the in sample data only. Create a chart that shows, out of sample:

- Benchmark (see definition above) normalized to 1.0 at the start: Green line
- Performance of manual strategy: Red line
- Both should be normalized to 1.0 at the start.

Create a table that summarizes the performance of the stock, and the manual strategy for both in sample and out of sample periods. Explain WHY these differences occur.

Part 5: Implement author() function (deduction if not implemented)

You should implement a function called `author()` that returns your Georgia Tech user ID as a string. This is the ID you use to log into Canvas. It is not your 9 digit student number. Here is an example of how you might implement author():

```
def author():
    return 'tb34' # replace tb34 with your Georgia Tech username.
```

Implementing this method correctly does not provide any points, but there will be a penalty for not implementing it.

Hints

Overall, I recommend the following steps in the creation of your strategies:

- Indicator design hints:
  - For your X values: Identify and implement at least 3 technical features that you believe may be predictive of future return.
- Rule based design:

- Use a cascade of if statements conditioned on the indicators to identify whether a LONG condition is met.
- Use a cascade of if statements conditioned on the indicators to identify whether a SHORT condition is met.
- The conditions for LONG and SHORT should be mutually exclusive.
- If neither LONG or SHORT is triggered, the result should be DO NOTHING.
- For debugging purposes, you may find it helpful to plot the value of the rule-based output (-1, 0, 1) versus the stock price.

Choosing Technical Features -- Your X Values

You should have already successfully coded the Bollinger Band feature:

```
bb_value[t] = (price[t] - SMA[t])/(2 * stdev[t])
```

Two other good features worth considering are momentum and volatility.

```
momentum[t] = (price[t]/price[t-N]) - 1
```

Volatility is just the stdev of daily returns.

It is usually worthwhile to standardize the resulting values (see https://en.wikipedia.org/wiki/Standard_score).

Contents of Report

Describe each indicator you use in sufficient **detail** that someone else could reproduce it. You should also provide a compelling description regarding why that indicator might work and how it could be used. You should also provide one or more charts that convey how each indicator works in a compelling way. (up to 8 charts).

For the best possible strategy, describe how you created it and any assumptions you had to make to make it work. Provide a chart that illustrates its performance versus the benchmark.

For your manual strategy, describe how you combined your indicators to create an overall signal. How do you decide to enter and exit your positions and why? Why do you believe (or not) that this is an effective strategy? Provide a chart.

Compare the performance of your manual strategy versus the benchmark for the in sample and out of sample time periods. Provide a chart.

Your report should be no more than 3000 words. Your report should contain no more than 14 charts. Penalties will apply if you violate these constraints.

Expectations

- In-sample backtests should perform very well.
- Out-of-sample backtests should... (you should be able to complete this sentence).

What to turn in

Turn your project in via Canvas.

- Your report as `report.pdf`
- All of your code, as necessary to run as `.py` files.
- Document how to run your code in `readme.txt`.
- No zip files please.

Rubric

Start with 100 points, deductions as follows:

Neatness (up to 5 points deduction if not).

Bonus for exceptionally well written reports (up to 2 points)

Indicators (up to 20 points potential deductions):

- Is there a compelling description why each indicator might work (-2 for each, up to a total of 6 off)
- Is each indicator described in sufficient detail that someone else could reproduce it? (-5 points for each if not)
- Is there a chart for each indicator that properly illustrates its operation? (-5 points for each if not)
- Is at least one indicator different from those provided by the instructor's code (i.e., another indicator that is not SMA, Bollinger Bands or RSI) (-10 points if not)
- Does the submitted code `indicators.py` properly reflect the indicators provided in the report (-20 points if not)

Theoretically optimal (up to 20 points potential deductions):

- Is the methodology described correct and convincing? (-10 points if not)
- Is the chart correct (dates and equity curve) (-10 points if not)
- Is the chart correct (dates and equity curve) (-10 points if not)
- Historic value of benchmark normalized to 1.0 with green line (-5 if not)
- Historic value of portfolio normalized to 1.0 with red line (-5 if not)
- Are the reported performance criteria correct ? See appropriate section for required stats. (-2 points for each item if not)

Manual rule-based trader (up to 50 points deductions):

- Is the trading strategy described with clarity and in sufficient detail that someone else could reproduce it? (-20)
- Does the provided chart(s) include:
  - Historic value of benchmark normalized to 1.0 with green line (-10 if not)
  - Historic value of portfolio normalized to 1.0 with red line (-10 if not)
  - Are the appropriate date ranges covered? (-10 if not)
  - Are vertical lines, appropriately colored, included to indicate entries (-10 if not)
- Does the submitted code `ManualStrategy.py` properly reflect the strategy provided in the report? (-20 if not)
- Does the submitted code and report reflect an understanding of the subject matter (up to -30 if not)
- Does the manual trading system provide higher cumulative return than the benchmark over the in-sample time period? (-10 if not)
- Did the student use the correct symbol? (-10 if not)
- Did the student use the date periods? (-10 if not)
- Does the strategy obey holding constraints (-5 if not)

Comparative analysis (up to 10 points deductions):

- Is the appropriate chart provided (-5 for each missing element, up to a maximum of -10)
- Are differences between the in-sample and out-of-sample performances appropriately explained (-5)

- Does the submitted code and report reflect an understanding of the subject matter (up to -5 if not)
- Is the required table present and correct (up to -5 if not)

Submission of code and report (up to 100 points deductions):

- Is the required code provided, including code to recreate the charts and usage of correct trades data frame. **DO NOT use plt.show() and manually save your charts. The charts should be created and saved using Python code** (up to -100 if not)
- Is the required report provided (-100 if not)

Required, Allowed & Prohibited

Required:

- Your project must be coded in Python 3.6.x.
- Your code must run on one of the university-provided computers (e.g. buffet01.cc.gatech.edu).
- Use only the API functions in util.py to read data.
- All charts must be generated in Python, and you must provide the code you used.

Allowed:

- You can develop your code on your personal machine, but it must also run successfully on one of the university provided machines or virtual images.
- Your code may use standard Python libraries.
- You may use the NumPy, SciPy, matplotlib and Pandas libraries. Be sure you are using the correct versions.
- Code provided by the instructor, or allowed by the instructor to be shared.
- A herring.

Prohibited:

- Generating charts using a method other than Python.
- Any other method of reading data besides util.py
- Modifying (or depending on modifications to) util.py.
- Any libraries not listed in the "allowed" section above.
- Any code you did not write yourself.

FAQs

- **Q:** I want to read some other values from the data besides just adjusted close, how can I do that? **A:** Look carefully at util.py and you will see that you can query for other values.

- **Q:** Are we only allowed one position at a time? **A:** You can be in one of three states: -1000 shares, +1000 shares, 0 shares.

- **Q:** Are we required to trade in only 1000 share blocks? (and have no more than 1000 shares long or short at a time? **A:** You can trade up to 2000 shares at a time as long as you maintain the requirement of holding 1000, 0 or -1000 shares.

- **Q:** Are we limited to leverage of 2.0 on the portfolio? **A:** There is no limit on leverage.

Legacy

- MC3-Project-2-Legacy-trader

- MC3-Project-2-Legacy
- MC3-Project-3-Legacy-Q
- MC3-Project-3-Legacy