CS 458/558 - Problem Set 3

Deadline: Wednesday November 4, 11:59pm

Algorithmic Trading: Momentum Strategy

There is a saying on Wall Street: buy low, sell high.

There is another saying on Wall Street: the trend is your friend.

In this assignment, we combine these two hypotheses by implementing and back testing a momentum trading strategy based on moving averages. Suppose you are interested in buying stock X. You calculate the average closing price of the stock for the past 50 trading days, which is $50. The current price of the stock is $49. Do you buy? No. You wait a day. You recalculate the 50 day moving average and get $49.50. Also, the new price of the stock is $50. Do you buy? Yes. When do you sell? When the 50 day moving average moves above your purchase price.

That's it.

Part 1: backtest()

Write a function `backtest` which takes the following parameters:

- ticker. A stock ticker, such as "AAPL". Default is "HD".
- start. The begining date for the back test period. Default is "2006-10-01 ".
- end. The ending date for the back test period. Default is "2015-10-01".
- duration. The length of the moving average period. Common durations are 20, 50, 100, and 200 days. Default is 50 days.

Your program should test the moving average trading strategy for the given stock for the given period and the given moving average duration. The output of the function is the investment return generatated by the strategy over the given period, e.g., 22%. (It should be a floating point number.)

You may use either R or Python. However, the R stockPortfolio package seems to be a good place to start.

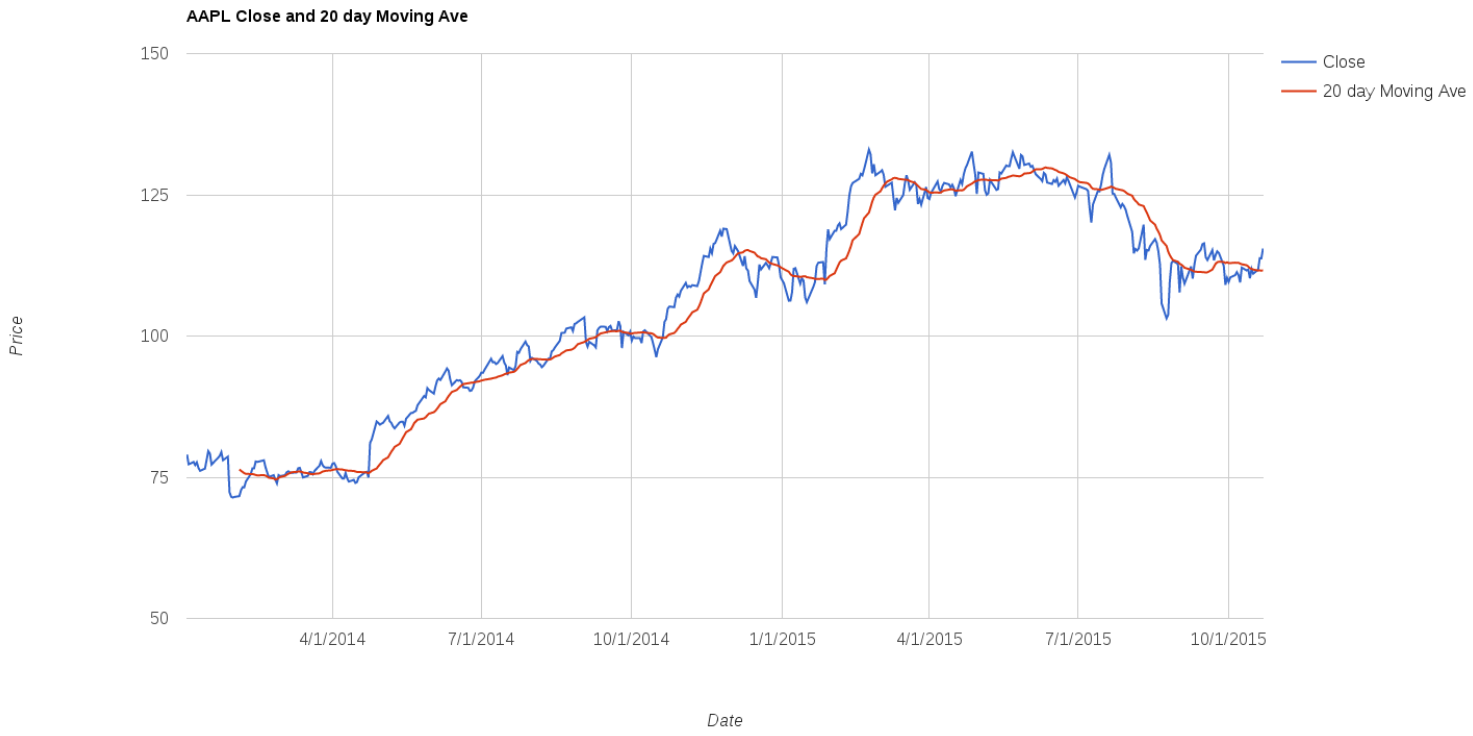
Spreadsheets are often used for this sort of model. Here is a Google spreadsheet of a 20 day moving average for Apple stock for almost two years.

For this example,

```
backtest("AAPL", "2014-01-01", "2015-10-23", 20) => 3.14% realized gains.
```

Note the difference between realized and unrealized gains.

AAPL Moving Average Chart



This spreadsheet uses the googlefinance() function, which is useful for analyzing equities and mutual funds.

Part 2: sectortest()

Each stock is part of a larger industry or sector. For example, Apple is usually in the technology sector. There are Exchange Traded Funds (ETF's) that comprise baskets of stocks for a given sector, yet trade like individual stocks. Their prices are quoted on the exchanges and can be bought and sold throughout the day. (Mutual funds, by contrast, can only be traded at the end of day price.)

For part 2, we consider the sector ETFs called SPDR's. For example, the technology SPDR ETF has the ticker

symbol XLK.

Using your `backtest` function from part 1, test each of the eleven ETFs, over multiple periods and multiple moving average durations. The `sectortest()` function should take two parameters:

- a list of (start, end) period tuples.
- a list of moving avarage durations.

The function should test all eleven sector ETFs for all combinations of periods and durations. It should return the best and worst performing result. It should also indicate which period had the highest average returns, and which duration had the highest average returns

For example, given three periods and three durations (20, 50, and 100 days), `sectortest` should perform 99 backtests, returning the best and worst result, as well as the best average period and duration.

The function should take an argument of file path, and save the output to the given file.

```
sectortest(file = PATH) => no return value.
Generate a file "PATH", containing the following
(everything is separated by a space)

best XLK 2009-01-01 2015-10-01 100 27.0
worst XLU 2003-01-01 2008-10-01 200 -37.0
avg-period 2009-01-01 2015-10-01 28.0
avg-duration 50 38.0
```

Part 3: realbacktest()

The `backtest` function described above ignored reality. In particular, each trade has a cost, usually a commission and exchange fees. This transaction cost can convert a winning strategy to a loser. In addition, the moving average strategy is not guaranteed to outperform a simple buy and hold strategy - particularly if you consider the total return that would include dividends. With buy and hold, you get paid every dividend during the holding period. With the

moving average strategy, you may sometimes find yourself not holding the stock on the date required to receive a dividend.

Revise your `backtest` function to take another parameter, `commission`, which is the number of basis points you pay every time you perform a transaction, either a buy or a sell. For example, if the commission is 2 basis points and the stock price is $80, you pay $.016 on the trade. This cost will reduce your overall performance.

You should also calculate the gain or loss of the buy and hold strategy. That is, what is your (unrealized) capital gain or loss if you simply bought the stock on day 1 of the period and held it until the end?

For up to 5 points of extra credit, calculate the total return of the investment which includes not only capital gains and transaction costs, but also any dividends received - both for moving average strategy and buy and hold. That is, the buy and hold strategy will receive every dividend in the period. The moving average strategy will possibly miss out on some dividends.

For this example,

```
realbacktest("AAPL", "2014-01-01", "2015-10-23", 20, 2, file = PATH) =>

the output file contains the following:

1.39% net return, moving average.
46.17% buy and hold return.
50.41% total buy and hold return.
```