

Course Project Part IV – 50pts

Part IV – Your first trading strategy – 50 pts

You will reuse the data from the part III.

Introduction

It will be represented by the following dictionary.

```
all_data[LIST_OF_SYMBOLS]
```

During the previous assignment we observed that Southwest has a lag of a few days. The goal of this part is to use this information to make money out of this information.

For this part we will be using a very famous signal for trading strategy.

From

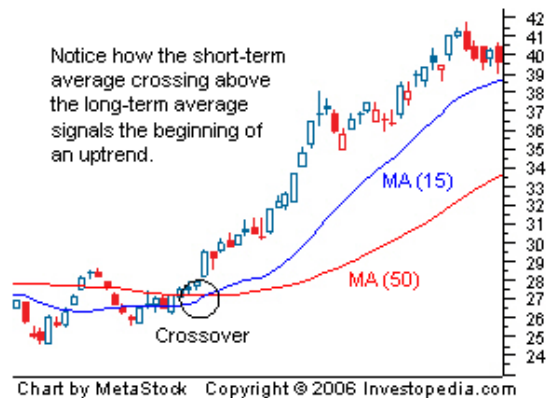
<http://www.investopedia.com/university/movingaverage/movingaverages4.asp>

Crossovers

A crossover is the most basic type of signal and is favored among many traders because it removes all emotion. The most basic type of crossover is when the price of an asset moves from one side of a moving average and closes on the other. Price crossovers are used by traders to identify shifts in momentum and can be used as a basic entry or exit strategy. As you can see in Figure 1, a cross below a moving average can signal the beginning of a downtrend and would likely be used by traders as a signal to close out any existing long positions. Conversely, a close above a moving average from below may suggest the beginning of a new uptrend.



The second type of crossover occurs when a short-term average crosses through a long-term average. This signal is used by traders to identify that momentum is shifting in one direction and that a strong move is likely approaching. A buy signal is generated when the short-term average crosses above the long-term average, while a sell signal is triggered by a short-term average crossing below a long-term average. As you can see from the chart below, this signal is very objective, which is why it's so popular.



Testing your theory

Task 1 (10pts):

In this part, you will try to exploit the lag from LUV to make money out of LUV.

- You will calculate the moving average (on adjusted price) over 25 days for LUV.
- You will calculate the moving average (on adjusted price) over 5 days for LUV.
- You will need to store the previous results into a dataframe where you will have the price for LUV, the daily return for Airline Industry (all the symbols except LUV and WTI), the moving average over 5 days for LUV, the moving average over 25 days for LUV

Task 2 (10pts): Creation of a signal

You will use the following statement: *A buy signal is generated when the short-term average crosses above the long-term average, while a sell signal is triggered by a short-term average crossing below a long-term average.*

You will create a new column *Signal* in the data frame indicating an order to buy (+1) and order to sell (-1).

Task 3 (10pts): Calculate the Profit and Loss

You will add a new column *PNL* to this dataframe.

Each time, you will have a +1 in the column *Signal*, you will add the price to the column *PNL* and each time you have a -1 in the column *Signal*, you will subtract the price to the column *PNL*.

The last row should give you the total *PNL* you got during the full period of this class project.

Task 4 (5pts): Can you improve the signal?

(No code on this part)

Since we have the information that LUV follows the Airline Industry movement, can you improve the previous signal? How would you code this part?

Task 5 (15pts): Create a class trading strategy

Once you study a model, you are going to implement a *trading_strategy* class. This *trading_strategy* will have a function *process_tick*(adusted_price). This function will get the adjusted price one by one. Inside this class, you will have two member variables *moving_average_25* and *moving_average_5*.

Each time you receive a tick, you will re-calculate the moving averages.

If you get a signal (when you have a cross), you will need to display the order.

The order will just be a string that you will print when you process a tick.

Example: "BUY {price} {date}" or "SELL {price} {date}".

In your code, you will have your class:

```
class trading_strategy:
    self.moving_average_25=0
    self.moving_average_5=0
    self.pnl=0
    ...
    def __init__(self):
    def process_tick(self, adusted_price):
    def check_signal (self, ...):
    def update_pnl (self, ...):
    def generate_buy_order (self, ...):
    def generate_sell_order (self, ...):
```

... is not a part of the python syntax. It means you need to figure out which parameters will be adequate to your code.

Each time you process a tick, you update your moving average, you check if you have a signal (meaning if you have a cross), depending on the result of this signal, you will call *generate_buy_order* or *generate_sell_order*.

The code of the *main* should be close to the following code.

```
ts1 = trading_strategy()

for adjusted_price in (you need to find what you need write here)
    ts1.process_tick(adjusted_price)
```

When you will run your code, you will update the PNL for each order you will get.

At the end of the execution, we will use:

```
ts1.display_pnl()
```

This function will be just display the final PNL you got with your strategy.

This number should be exactly the same as the one you found when you studied the strategy in the Task 3.