# Building a Simple Trading Strategy in Python—Trend Following | by Bee Guan Teo | in The Handbook of Coding in Finance

Trading strategy is essential in any investment. **Python** offers us a fantastic opportunity to test and implement our trading idea based on historical data and observe the result. This helps us to evaluate the feasibility of our trading strategy before we adopt it in the real market.

In this article, we are going to walk through how we can use Python to implement and test a simple trading strategy based on a **trend following approach**.

> Trend following or trend trading is a trading strategy according to which one should buy an asset when its price trend goes up, and sell when its trend goes down, expecting price movements to continue. [Source: Wikipedia]

***Disclaimer****: The writing of this article is only aimed at demonstrating the steps to implement a trading strategy in Python. It doesn't serve any purpose of promoting any stock or giving any specific investment advice.*

## Prerequisite Python Packages

1. **yFinance** — https://pypi.org/project/yfinance/
2. **Numpy** — https://numpy.org/
3. **Matplotlib** — https://matplotlib.org/
4. **Pandas** — https://pandas.pydata.org/

## Github

The original full source codes presented in this article are available on my Github Repo. Feel free to download it (*TrendFollowing.ipynb*) if you wish to use it to follow my article.

## Building Trend Following Strategy

### 1. Acquisition of stock data

Firstly, we are going to use *yFinance* to obtain the stock data. yFinance is an open-source Python library that allows us to acquire stock data from Yahoo Finance without any cost.

Here, we are going to acquire the stock prices of *GOOGL* over one year.

**Line 1–4:** Import all the necessary libraries

**Line 7–8:** Use the *yFinance* download method to acquire stock prices of *GOOGL* over the entire year of 2021.

| Date | Open | High | Low | Close | Adj Close | Volume |
|---|---|---|---|---|---|---|
| 2021-01-04 | 1760.000000 | 1762.489990 | 1707.140015 | 1726.130005 | 1726.130005 | 1866200 |
| 2021-01-05 | 1725.089966 | 1746.829956 | 1716.900024 | 1740.050049 | 1740.050049 | 1018000 |
| 2021-01-06 | 1700.260010 | 1743.969971 | 1696.099976 | 1722.880005 | 1722.880005 | 2329400 |
| 2021-01-07 | 1726.760010 | 1777.819946 | 1726.760010 | 1774.339966 | 1774.339966 | 2096800 |
| 2021-01-08 | 1777.160034 | 1799.359985 | 1761.219971 | 1797.829956 | 1797.829956 | 1774200 |

Image Prepared by the Author

**2. Generating Fast Signal and Slow Signal**

We need to generate signals to determine if the current stock price trend goes up or down. Two types of signals are required for our strategy:

- Fast signal — Moving average of stock prices over recent 10 days (MA-10)
- Slow signal — Moving average of stock prices over recent 50 days (MA-50)

**Line 1–2:** Use the Pandas rolling method to calculate MA-10 and MA-50, respectively.

**Line 3:** Drop the rows with null values.

| Date | Open | High | Low | Close | Adj Close | Volume | MA10 | MA50 |
|---|---|---|---|---|---|---|---|---|
| 2021-03-16 | 2065.989990 | 2113.679932 | 2059.290039 | 2083.889893 | 2083.889893 | 1595000 | 2051.532996 | 1959.137598 |
| 2021-03-17 | 2068.469971 | 2099.000000 | 2044.119995 | 2082.219971 | 2082.219971 | 1319100 | 2058.613989 | 1966.259397 |
| 2021-03-18 | 2048.179932 | 2068.750000 | 2019.180054 | 2021.339966 | 2021.339966 | 1585000 | 2057.354980 | 1971.885195 |
| 2021-03-19 | 2029.729980 | 2037.040039 | 2002.930054 | 2026.959961 | 2026.959961 | 2303600 | 2050.343970 | 1977.966794 |
| 2021-03-22 | 2027.630005 | 2048.340088 | 2014.000000 | 2030.689941 | 2030.689941 | 1676800 | 2052.662964 | 1983.093794 |

Image Prepared by the Author

If the fast signal is larger than slow signal, the stock price is expected to go up over the next several days. Otherwise, the price will go down.

**3. Generating long positions**

Our strategy is developed based on the rationale below:

**If MA-10 > MA-50, we buy and hold one share of stock.**

This means we are going to enter a long position whenever MA-10 > MA-50.

**Line 1–2:** Set two possible conditions: MA-10 > MA-50 or MA-10 < MA-50.

**Line 5:** Define value of choices.

**Line 8:** Use the *Numpy selec*t method generate a number series of 1 and 0 based on conditions that if MA-10 > MA-50, set value as 1, else set value as 0. The value of 1 denotes a long position.

| Date | Open | High | Low | Close | Adj Close | Volume | MA10 | MA50 | Action | Position |
|---|---|---|---|---|---|---|---|---|---|---|
| **2021-03-16** | 2065.989990 | 2113.679932 | 2059.290039 | 2083.889893 | 2083.889893 | 1595000 | 2051.532996 | 1959.137598 | 1 | 1 |
| **2021-03-17** | 2068.469971 | 2099.000000 | 2044.119995 | 2082.219971 | 2082.219971 | 1319100 | 2058.613989 | 1966.259397 | 1 | 1 |
| **2021-03-18** | 2048.179932 | 2068.750000 | 2019.180054 | 2021.339966 | 2021.339966 | 1585000 | 2057.354980 | 1971.885195 | 1 | 1 |
| **2021-03-19** | 2029.729980 | 2037.040039 | 2002.930054 | 2026.959961 | 2026.959961 | 2303600 | 2050.343970 | 1977.966794 | 1 | 1 |
| **2021-03-22** | 2027.630005 | 2048.340088 | 2014.000000 | 2030.689941 | 2030.689941 | 1676800 | 2052.662964 | 1983.093794 | 1 | 1 |

Image Prepared by the Author

**4. Calculating Daily Profit**

We can now calculate the daily profit for our long positions. The daily profit is equal to the close price of following day minus the close price of current day. We can then plot a line chart to visualize our daily profit.

**Line 1:** Use the Pandas shift method to obtain the closing price of the following days.

**Line 2:** Use the Python list comprehension to calculate the daily profit by subtracting the close price of next day with close price of today if the "Position" is equal to 1 (long position).

**Line 3–4:** Generate line chart to visualize the daily profit.

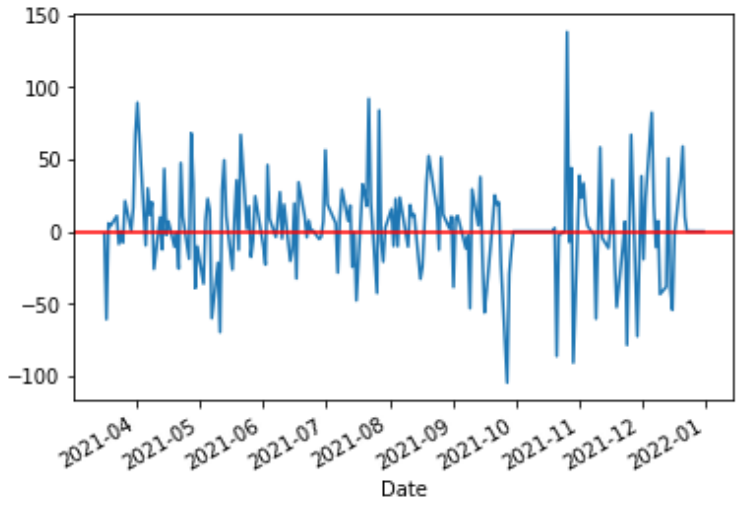


Image Prepared by the Author

We can observe that the daily profit can be positive, negative or zero. The positive denotes a profit gain whereas the negative denotes the loss. On the other hand, zero just means we do not have stock at hand and therefore the profit is zero.

**5. Calculating Cumulative Profit**

Instead of daily profit, we can also compute cumulative profit over the days and plot a line chart to visualize the profit.

**Line 1:** Use the *Pandas cumsum* method to calculate the cumulative profit over days.

**Line 2:** Plot a line chart to visualize the cumulative profit.

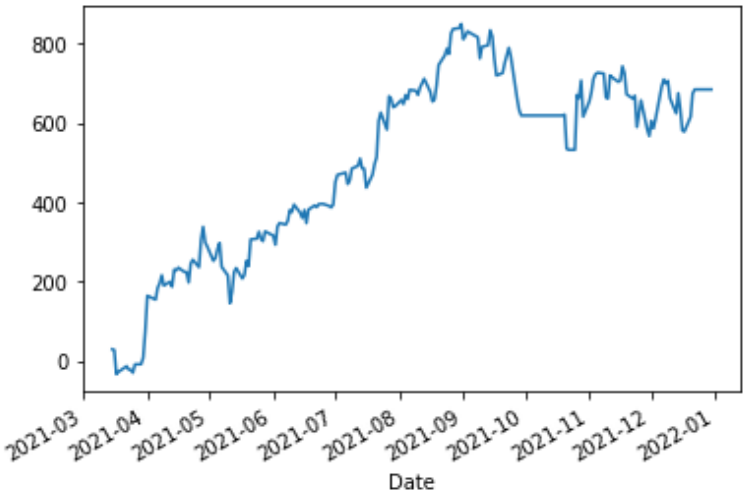


Image Prepared by the Author

In overall, we can observe our strategy shows a profit gain throughout the year with some fluctuation.

## Conclusions

Although the trend following strategy presented here is promising, it is merely a simple demonstration of showing how we can use Python to implement a trading idea. There are many more aspects to take into consideration before applying it in real market investment.

Hopefully, this article can help you to kickstart your journey to use Python in testing your trading idea.

I hope you enjoy reading this article.

## Subscribe to Medium

***If you like my article and would like to read more similar articles from me or other authors, feel free to subscribe to Medium. Your subscription fee will partially go to me. This can be a great support for me to produce more articles that can benefit the community.***

## Reference

1. https://en.wikipedia.org/wiki/Trend_following