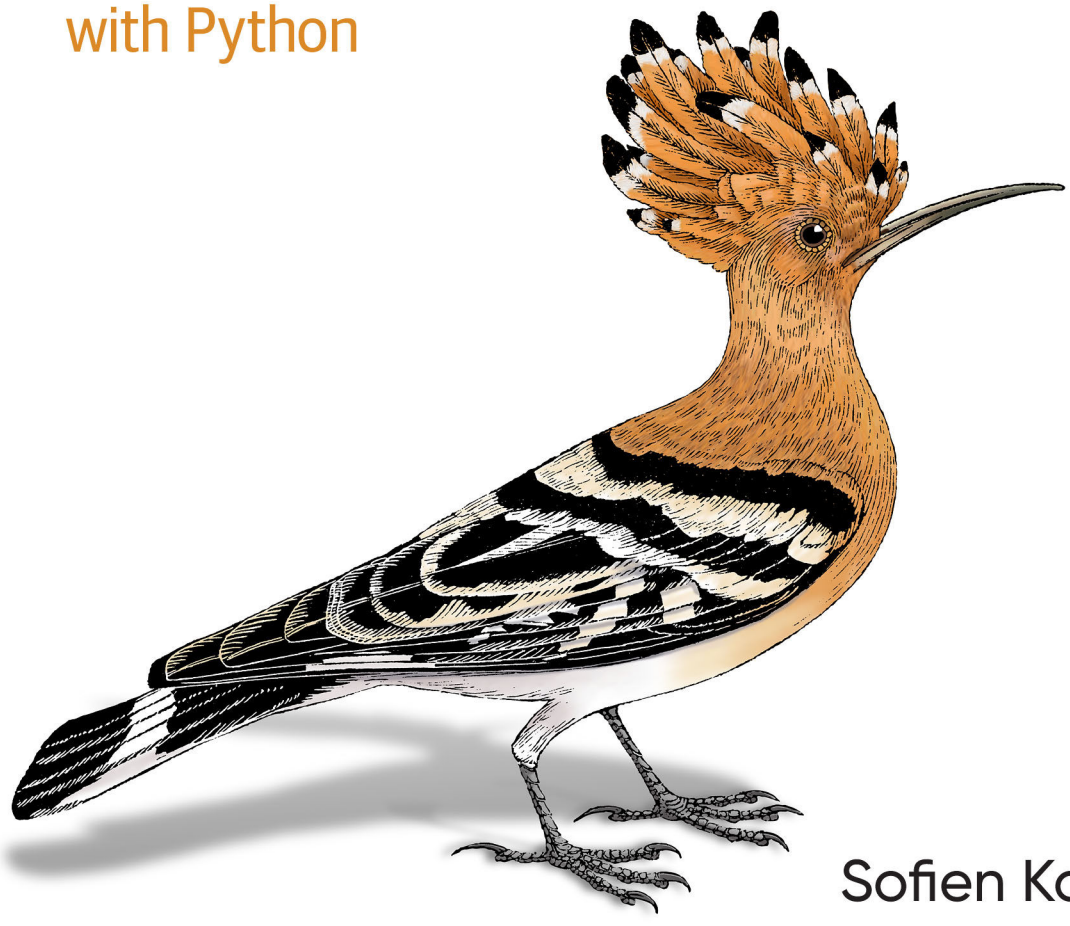
O'REILLY®

Mastering Financial Pattern Recognition

Finding & Back-Testing Candlestick Patterns with Python



Sofien Kaabar

Praise for *Mastering Financial Pattern Recognition*

Very well written. The book makes complicated stuff an easy read and is a must if you are passionate about trading.

—Saby Upadhyay, CEO, White Swan Global Markets

An extremely useful book for those interested in technical analysis, especially candlestick analysis. Easy to read and follow. I highly recommend this book for beginners and veterans alike.

—Sattam Al-Sabah, president, OneUp Trader

Provides a wealth of information on technical analysis and candlestick charting. Having the Python code as a starting point helps the reader implement and understand things that much faster.

—Timothy M. Kipper, Jr., founder, JDTK Investments

A comprehensive guide for trading using candlestick pattern-based signals, with detailed illustrations of indicators, easily implementable code, and a thorough overview of building your own systematic trading strategy.

—Ning Wang, quantitative investment strategies structurer, Barclays

Mastering Financial Pattern Recognition

Finding and Back-Testing Candlestick Patterns with Python

Sofien Kaabar

Beijing • Boston • Farnham • Sebastopol • Tokyo

O'REILLY®

Mastering Financial Pattern Recognition
by Sofien Kaabar

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Michelle Smith
Development Editor: Corbin Collins
Production Editor: Elizabeth Faerm
Copyeditor: Kim Wimpsett
Proofreader: Paula L. Fleming

Indexer: Potomac Indexing, LLC
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Kate Dullea

October 2022: First Edition

Revision History for the First Edition
2022-10-18: First Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781098120474* for release details.

978-1-098-12047-4

[LSI]

Table of Contents

Preface

> Finding patterns is the essence of wisdom.
>
> —Dennis Prager

With technological progress and the decentralization of financial information, coding and automated research have become integral parts of the trading world. Anyone who masters the art of trading and coding has a tremendous edge in the markets.

Trading techniques are numerous, and they can be based on many tools and concepts. For example, fundamental traders rely on economic and political analyses to derive a long-term view on different assets, while technical traders rely on more quantitative measures and a few psychological concepts to forecast the next likely moves of the markets.

Therefore, we can say that on a high level there exist two types of analyses, fundamental and technical. This book will present in detail a field in technical analysis called *candlestick pattern recognition*.

Why This Book?

I have spent my career researching trading strategies, patterns, and anything related to the financial world. I have developed a special passion for patterns, and specifically candlestick patterns, due to their widespread adoption but also their interesting performance results. Moreover, throughout the years, I have discovered a few candlestick patterns that I believe can at least rival the classical patterns. This brings us to the purpose of writing the book: I am aiming to present the totality of candlestick patterns, including my personal ones, and how to code a system that back-tests them across a wide variety of markets.

Machines can perform pattern recognition and detection better than humans because of their objectivity. Therefore, I have dedicated the first chapters of the book to creating the structure of a candlestick pattern recognition algorithm before moving on to

dig deep into patterns and strategies in the later chapters. This means that the first skill you will learn is how to automate the data import process in Python.

There are many classical candlestick patterns, and it is everyone's duty to test them to see whether they actually are predictive. After all, if we use these patterns to forecast the markets, we should have objective results that prove they are indeed value-adders. We will get such results and interpret them, just as I do with the candlestick patterns that I have discovered over the years. We will also see the advantages and limitations of every pattern.

When we do find the good patterns that help with the predictive task, you should insert them in the overall trading framework, which includes other tools and a risk management system. You will learn how to code technical indicators and combine them with candlestick patterns to create trading signals. Finally, you will back-test these signals, and you will be able to optimize the parameters so that you get a good full-scale pattern recognition strategy.

Hence, the utility of this book is to show you how to automate your research by letting the algorithms you create evaluate the different candlestick patterns. Finally, you will learn how to determine your strategy, which will use the patterns and combine with other technical indicators.

Target Audience

This book is suited to aspiring students, academics, curious minds, and finance practitioners who are interested in candlestick pattern recognition and its applications in finance. You will benefit from this book if you are interested not only in using Python but also in developing strategies and technical indicators.

The book assumes you have basic background knowledge in both Python programming (professional Python users will find the code very straightforward) and financial trading. I take a clear and simple approach that focuses on the key concepts so that you understand the purpose of every idea.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, email addresses, filenames, and file extensions.

`Constant width`
: Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

`Constant width bold`
: Shows commands or other text that should be typed literally by the user.

`Constant width italic`
: Shows text that should be replaced with user-supplied values or by values determined by context.

This element signifies a tip or suggestion.

This element signifies a general note.

This element indicates a warning or caution.

Using Code Examples

Supplemental material (code examples, exercises, etc.) is available for download at *https://github.com/sofienkaabar/mastering-financial-pattern-recognition*.

If you have a technical question or a problem using the code examples, please send email to *bookquestions@oreilly.com*.

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but generally do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Mastering Financial*

Pattern Recognition by Sofien Kaabar (O'Reilly). Copyright 2023 Sofien Kaabar, 978-1-098-12047-4."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

O'Reilly Online Learning

For more than 40 years, *O'Reilly Media* has provided technology and business training, knowledge, and insight to help companies succeed.

Our unique network of experts and innovators share their knowledge and expertise through books, articles, and our online learning platform. O'Reilly's online learning platform gives you on-demand access to live training courses, in-depth learning paths, interactive coding environments, and a vast collection of text and video from O'Reilly and 200+ other publishers. For more information, visit *https://oreilly.com*.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at *https://oreil.ly/mstrg-finan-pttrn-recog*.

Email *bookquestions@oreilly.com* to comment or ask technical questions about this book.

For news and information about our books and courses, visit *https://oreilly.com*.

Find us on LinkedIn: *https://linkedin.com/company/oreilly-media*.

Follow us on Twitter: *https://twitter.com/oreillymedia*.

Watch us on YouTube: *https://youtube.com/oreillymedia*.

Acknowledgments

Nothing would be the same without the support of my parents, which is why I can't help but acknowledge their direct and indirect impact on the book. I would also like to mention that none of this would be possible without the patience of my wife, Charline, who tolerated my late-night writing. I have nothing but gratitude toward her.

I would also like to acknowledge the debt I owe to the editors, Michelle Smith and Corbin Collins, as well as production editor Elizabeth Faerm, for their continued support and the amazing job they do and also for their patience. Similarly, I would like to thank every person involved at O'Reilly Media.

Additionally, my special thanks go to the great technical reviewers, Ning Wang, Timothy Kipper, and Kushan Vora, for their immense contributions. They have had a sizable impact on making this book readable, useful, and straightforward. I could not ask for better people to review my book.

Finally, I am deeply grateful to you, the reader, for investing your time into reading my work and for placing your trust in my research. I hope you find it useful.

CHAPTER 1

Candlestick-Based Trend-Following Strategies

Pattern recognition is only one part of the equation. When you a find a pattern, you are most likely to use it within a wider trading framework, as trading systems relying purely on candlestick patterns are unlikely to yield consistent and stable positive returns.

This chapter discusses some examples of strategies you can use to filter the patterns using specific technical rules when dealing with trend following. Each market has its own characteristics and must have optimized strategy parameters, which is why I recommend you focus on understanding the main ideas and concepts rather than take away the exact parameters of the strategies presented in this chapter.

Ideally, the strategies should help you understand how to combine the different candlestick patterns (classic and modern) with some technical indicators. I also discuss the indicators in this chapter so that you understand their mechanisms and their weaknesses.

Filtering refers to the concept of choosing the signals that have a better probability of providing a positive return.

Combining the Double Trouble Pattern with the RSI

As a reminder, the Double Trouble pattern is a modern trend-following candlestick configuration that incorporates volatility in its characteristics (calculated using the ATR indicator). The RSI is an indicator that, as discussed previously in the book, reinterprets the market's momentum into values between 0 and 100 to better understand the current dynamic.

Even though the RSI is a contrarian indicator, I will show you a technique for using it within a trend-following framework.

This strategy uses the RSI as a filter for the detected Double Trouble patterns. This means that whenever a pattern is found, it is run through an RSI filter to validate the signal.

The RSI can also be a momentum gauge. Therefore, values above 50 represent bullish momentum (uptrend) while values below 50 represent bearish momentum (downtrend). This is one way to use the RSI as a trend-following indicator.

The trading conditions of the strategy are as follows:

- A long signal is generated whenever a bullish Double Trouble pattern appears while the 14-period RSI is above 50.
- A short signal is generated whenever a bearish Double Trouble pattern appears while the 14-period RSI is below 50.

The filter is therefore the value of the RSI relative to the neutrality level of 50. This is used to increase the hit ratio of the trades using the invisible hand mechanism.[1] The basic intuition of the strategy is to take only bullish signals in a bullish regime and only bearish signals in a bearish regime.

The following code snippet shows how to code the signal function of the strategy:

```
def signal(data, open_column, high_column, low_column, close_column,
           atr_column, rsi_column, buy_column, sell_column):

    data = add_column(data, 5)

    for i in range(len(data)):

        try:
```

1 The *invisible hand mechanism* involves using trades in tandem with the trend, for example, only taking buy signals in a bullish trend while disregarding any sell signals.

```
            # Bullish setup
            if data[i, close_column] > data[i, open_column] and \
               data[i, close_column] > data[i - 1, close_column] and \
               data[i - 1, close_column] > data[i - 1, open_column] and \
               data[i, high_column] - data[i, low_column] > \
               (2 * data[i - 1, atr_column]) and \
               data[i, close_column] - data[i, open_column] > \
               data[i - 1, close_column] - data[i - 1, open_column] and \
               data[i, buy_column] == 0 and \
               data[i, rsi_column] > 50:

                     data[i + 1, buy_column] = 1

            # Bearish setup
            elif data[i, close_column] < data[i, open_column] and \
               data[i, close_column] < data[i - 1, close_column] and \
               data[i - 1, close_column] < data[i - 1, open_column] and \
               data[i, high_column] - data[i, low_column] > \
               (2 * data[i - 1, atr_column]) and \
               data[i, open_column] - data[i, close_column] > \
               data[i - 1, open_column] - data[i - 1, close_column] and \
               data[i, sell_column] == 0 and \
               data[i, rsi_column] < 50:

                     data[i + 1, sell_column] = -1

      except IndexError:

          pass

   return data
```

Notice the addition of the filter through the RSI line of code stating its value relative to the neutrality level 50. Figure 1-1 shows a signal chart on USDCAD.

You can always tweak the strategy so that it better fits your profile and the market's characteristics. However, you should keep in mind that the RSI is not a miraculous predictor of trend as it is price-derived and therefore lagging (that is, it does not predict the future but simply tells the story of the past). Also, the RSI tends to break and reintegrate the 50 level many times, which may give faulty signals.

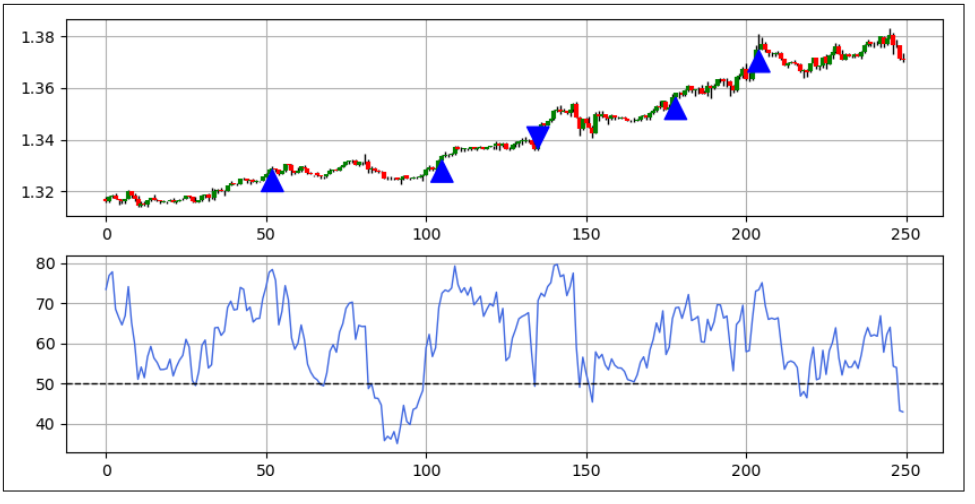


Figure 1-1. Signal chart of the strategy using the Double Trouble pattern and the RSI

Combining the Three Candles Pattern with Moving Averages

Moving averages are great trend filters and help in determining whether to take the signal or not. Remember, a *moving average* is a mean that follows the market price using a rolling window. This strategy uses two trend-following elements from different fields[2] in order to give a signal.

The Three Candles pattern is a trend-following configuration composed of big-bodied homogenous successive candlesticks that confirm the underlying trend. Generally, a bullish Three Candles pattern is called Three White Soldiers, while the bearish version is called Three Black Crows.

Similar to using the RSI as a trend follower, using moving averages is extremely useful in determining the underlying trend.

When you see a market above its moving average (for example, a 100-period moving average), it is generally a sign that a bullish regime is in progress, and when you see a

2 The Three Candles pattern comes from the field of pattern recognition, while moving averages come from the field of statistics and trend-following technical indicators.

market below its moving average, it is generally a sign that a bearish regime is in progress, thus indicating which type of signal to prefer.

The trading conditions of the strategy are as follows:

- A long signal is generated whenever a Three White Soldiers pattern appears while the market price is above its 100-period moving average.
- A short signal is generated whenever a Three Black Crows pattern appears while the market price is below its 100-period moving average.

The trend-following strategies use a combination of a candlestick pattern and a trend filter that increases the conviction in the trade. Although historical back-testing shows that the hit ratio is not always increased with the filter, for some markets, the ratio is significantly enhanced.

The following code snippet shows how to code the signal function of the strategy:

```
def signal(data, open_column, close_column, ma_column, buy_column,
           sell_column):

    data = add_column(data, 10)

    for i in range(len(data)):

       try:

           # Bullish setup
           if data[i, close_column] - data[i, open_column] > body and \
              data[i - 1, close_column] - data[i - 1, open_column] > \
              body and data[i - 2, close_column] - \
              data[i - 2, open_column] > body and data[i, close_column] \
              > data[i - 1, close_column] and data[i - 1, close_column] \
              > data[i - 2, close_column] and data[i - 2, close_column] \
              > data[i - 3, close_column] and data[i, close_column] > \
              data[i, ma_column] and data[i, buy_column] == 0:

                    data[i + 1, buy_column] = 1

           # Bearish setup
           elif data[i, open_column] - data[i, close_column] > body and \
                data[i - 1, open_column] - data[i - 1, close_column] > body \
                and data[i - 2, open_column] - data[i - 2, close_column] \
                > body and data[i, close_column] < \
                data[i - 1, close_column] and data[i - 1, close_column] \
                < data[i - 2, close_column] and data[i - 2, close_column] \
                < data[i - 3, close_column] and data[i, close_column] < \
                data[i, ma_column] and data[i, sell_column] == 0:
```

```
                data[i + 1, sell_column] = -1
        except IndexError:
            pass
    return data
```

Figure 1-2 shows an example of a signal chart with a 100-period moving average (the swooping line starting at the upper left), which acts as a filter. Notice how you see only bullish signals when the market is above its moving average and see only bearish signals when the market is below its moving average.

Figure 1-2. Signal chart of the strategy using the Three Candles pattern and moving averages

The strategy is best used in confirmation of other trend-following strategies. For example, a fundamental trader has a carry trade in USDJPY and sees that a Three White Soldiers pattern has emerged while the market is above its 100-period moving average. This observation can serve as a conviction enhancer or a confirmation to add more to the position. After all, trading is a numbers game, and stacking up the odds on your side increases the probability of gain.

A *fundamental trader* is a trader who relies on economic and financial analysis instead of technical analysis to make their decisions.

A *carry trade* is a currency position that entails going long on (buying) the currency with the higher interest rate and shorting (selling) the currency with the lower interest rate to benefit from the interest rate differential.

Combining the Bottle Pattern with the Stochastic Oscillator

Similar to the RSI, the stochastic oscillator is a momentum indicator heavily used in technical trading and known both to the retail and professional communities. The *stochastic oscillator* uses a basic normalization function to trap the values between 0 and 100. It is easier to calculate than the RSI. Before discussing the oscillator, let's look at the concept of *normalization*.

Whenever you have an array of different unbounded values such as the market price (or any other random set of time series), you can normalize the values between 0 and 1, with 0 being the lowest value for a specific time window and 1 being the highest value for a specific time window. Take a look at this table:

Time step	1	2	3	4	5
Value	10	40	5	90	25
Normalized value	0.06	0.41	**0.00**	**1.00**	0.24

The table shows that when the variable moves in time from period 1 to period 5, the normalized values show that they can be bounded between 0 and 1, with 0 referring to the lowest value (5) and 1 referring to the highest value (90). Similarly, notice how the value that lies approximately between 0 and 1 has a normalized value of 0.41, which means it is probably the middle value. This is normal as 40 is around half the distance between 5 and 90.

The following formula shows how to normalize any value between 0 and 1 given a specific time window:

$$x_{\text{Normalized}} = \frac{x_{\text{Original}} - x_{\text{Low}}}{x_{\text{High}} - x_{\text{Low}}}$$

Let's try an example using the previous table. Take the value 10 and try to normalize it using the formula. You should have the following calculation:

$$x_{\text{Normalized}} = \frac{10 - 5}{90 - 5} = 0.06$$

The stochastic oscillator normalizes the market price in a modified way by incorporating the highs and lows in the formula so that it becomes as follows:

$$\text{Stochastic value}_i = \frac{\text{Close}_i - \text{Low}_{i-n:i}}{\text{High}_{i-n:i} - \text{Low}_{i-n:i}}$$

The formula means that the current value of the stochastic is the difference between the current close price minus the lowest low price for a set lookback period divided by the difference between the highest high price and the lowest low price for the same lookback period.

The difference between the simple normalization function and the stochastic oscillator's function is the addition of the high price and low price to the latter.

The stochastic oscillator can be described as the smoothed version of the previous formula, and it is generally charted with a short-term moving average calculated on its value, called the *signal line*. To create the default stochastic oscillator, follow these steps:

1. Normalize the values using the stochastic oscillator's function using a rolling window of 14 periods.
2. Smooth out the result from the first step with a three-period moving average. This is the stochastic oscillator.
3. Calculate the signal line, which is another three-period moving average calculated on the values of the second step. This is the signal line.

The first moving average calculated on the stochastic oscillator is referred to as *smoothing*, while the signal line is referred to as *slowing*.

Similar to the RSI, the stochastic oscillator is bounded between 0 and 100 and has an oversold zone below 20 and an overbought zone above 80. Because of its formula, it is relatively more volatile than the RSI and tends to move from one extreme to another faster.

Many techniques are possible with the stochastic oscillator, but the one that interests us is the cross between it and its signal line. This is referred to as the *cross technique*

and is famous in contrarian strategies. (However, I am using it with a trend-following candlestick pattern and therefore turning it into a trend-following technique.)

You must be careful with the stochastic oscillator as it tends to stick to the extremes from time to time due to the nature of the normalization function, thus providing false signals. The stickiness effect is manifested when the oscillator remains in the oversold and overbought zones for extended periods of time.

The following snippet shows how to code the stochastic oscillator:

```
def stochastic_oscillator(data,
                          lookback,
                          high,
                          low,
                          close,
                          position,
                          slowing = False,
                          smoothing = False,
                          slowing_period = 1,
                          smoothing_period = 1):

    data = add_column(data, 1)

    for i in range(len(data)):

        try:

            data[i, position] = (data[i, close] - min(data[i - lookback \
                                + 1:i + 1, low])) / (max(data[i - lookback\
                                + 1:i + 1, high]) - min(data[i - lookback \
                                + 1:i + 1, low]))

        except ValueError:

            pass

    data[:, position] = data[:, position] * 100

    if slowing == True and smoothing == False:

        data = ma(data, slowing_period, position, position + 1)

    if smoothing == True and slowing == False:

        data = ma(data, smoothing_period, position, position + 1)

    if smoothing == True and slowing == True:

        data = ma(data, slowing_period, position,   position + 1)
```

```
            data = ma(data, smoothing_period, position + 1, position + 2)

        data = delete_row(data, lookback)

        return data
```

The trading conditions of the strategy are as follows:

- A long signal is generated whenever a bullish Bottle pattern appears while the stochastic oscillator is above its signal line.
- A short signal is generated whenever a bearish Bottle pattern appears while the stochastic oscillator is below its signal line.

The following code snippet shows how to code the signal function of the strategy:

```
    def signal(data, open_column, high_column, low_column, close_column,
               stochastic_column, signal_column, buy_column, sell_column):

        data = add_column(data, 5)

        for i in range(len(data)):

            try:

                # Bullish setup
                if data[i, close_column] > data[i, open_column] and \
                   data[i, open_column] == data[i, low_column] and \
                   data[i - 1, close_column] > data[i - 1, open_column] and \
                   data[i, open_column] < data[i - 1, close_column] and \
                   data[i, stochastic_column] > data[i, signal_column] and \
                   data[i, buy_column] == 0:

                        data[i + 1, buy_column] = 1

                # Bearish setup
                elif data[i, close_column] < data[i, open_column] and \
                     data[i, open_column] == data[i, high_column] and \
                     data[i - 1, close_column] < data[i - 1, open_column] and \
                     data[i, open_column] > data[i - 1, close_column] and \
                     data[i, stochastic_column] > data[i, signal_column] and \
                     data[i, sell_column] == 0:

                        data[i + 1, sell_column] = -1

            except IndexError:

                pass

        return data
```

Figure 1-3 shows the signal chart on USDCHF.

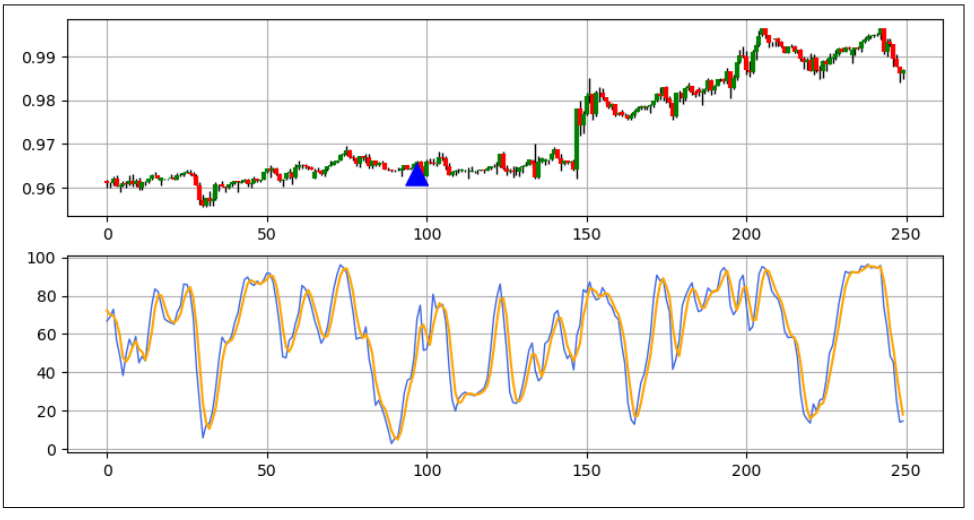


Figure 1-3. Signal chart of the strategy using the Bottle pattern and the stochastic oscillator

Combining the Marubozu Pattern with K's Volatility Bands

Let's get to square one with the first candlestick pattern covered in this book. The Marubozu pattern can be considered the most powerful candlestick since it has no wicks, meaning that the market went straight from one point to another without hesitation.

This strategy uses a concept known as a *volatility band*, which is a framing technique that envelops the market price to deliver dynamic support and resistance levels.

There are many types of volatility bands. The most well-known are Bollinger bands. The reliability of different types of volatility bands depends on the underlying markets and their parameters.

Before you can grasp K's volatility bands, you have to understand the basics of Bollinger bands. Developed by John Bollinger, the bands are more statistical in nature than technical.

Consider the following list: {11, 4, 5, 20}. Given the four values, how would you describe the elements? Generally, the best measure that describes elements in a list is

the mean. It is also the best estimate of the next expected value (if you were to add a new element chronologically). To calculate the mean of a list, follow this formula:

$$\chi = \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right) = \frac{x_1 + x_2 + \ldots + x_n}{n}$$

Therefore, according to the formula, you must sum them and divide the result by their quantity:

$$\chi = \frac{11 + 4 + 5 + 20}{4} = 10$$

Hence, the mean of the list is 10. Remember the concept of volatility from previous chapters where you use the ATR to approximate the fluctuations of the price relative to a past period. The bands use another technique to calculate volatility, which is the one used in descriptive statistics, that is, standard deviation.

Standard deviation is the square root of the squared deviations of each variable from the group's mean. The concept may sound complicated, but let's simplify it in a few steps:

- Calculate the distance of each variable (the close price) from the mean of the lookback period at the same time step.
- Square the distances so that you do not get negative values.
- Calculate the mean of these squared distances. The result is called the *variance*.
- Calculate the square root of the variance. The result is called the *standard deviation*.

Taking the square root in the last step allows you to compare apples to apples with the mean.

Mathematically speaking, the formula of the standard deviation is expressed as follows:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \chi\right)^2}$$

To purchase this book, you can visit O'Reilly's Website or Amazon

Click here to view on O'Reilly Media's Website

Click here to view on Amazon