

OptionLab: A Python library for evaluating option trading strategies



Roberto Gomes, PhD · [Follow](#)

4 min read · Jul 14, 2023



Listen



Share



Photo by cottonbro studio.

*NOTE: The library has undergone significant changes, rendering the code from the previous version of this article incompatible. As a result, this article has been updated to reflect the latest version of **OptionLab**.*

This is my first public Python library focused on computational finance. As the name suggests, **OptionLab** aims to be a tool used to evaluate option trading

strategies for their profit potential and associated risks. The world of options trading can be intricate and challenging, requiring a deep understanding of mathematical models and quantitative analysis. By leveraging Python's ease of use and flexibility, **OptionLab** simplifies the process of modeling even the most complex option strategies with just a few lines of code.

The library can be downloaded from the [project's GitHub page](#) or easily installed using **pip**:

```
pip install optionlab
```

If you want to support this and other open source projects that I maintain, become a [sponsor on Github](#).

An **OptionLab** run can produce several interesting outputs, such as:

- Profit/loss profile of the strategy on a user-defined target date,
- Range of stock prices for which the strategy is profitable,
- The Greeks associated with each leg of the strategy,
- The resulting debit or credit on the trading account,
- The maximum and minimum returns within a specified lower and higher price range of the underlying asset, and
- An estimate of the strategy's probability of profit, expected profit and expected loss.

The probability of profit (PoP) for the strategy is calculated based on the distribution of prices of the underlying asset on the user-defined target date. Specifically, for the price range in the payoff where the strategy generates profit, the PoP represents the probability that the stock price will fall within that range. The distribution of underlying asset prices on the target date can be derived from the Black-Scholes model (default) or obtained through simulations (e.g., Monte Carlo), or generated using machine learning models.

Basic documentation (I know it has to be improved!) is also available on the [project's GitHub page](#), along with examples of modeling common option trading strategies provided as Jupyter notebooks. I plan to add more examples in the near future (if I have enough time).

GitHub - rgaveiga/optionlab: A Python library for evaluating option trading strategies.

A Python library for evaluating option trading strategies. -
rgaveiga/optionlab

github.com

Despite the code having been developed with option strategies in mind, it can also be used for strategies that combine options with stocks, such as the strategy known as [covered call](#).

In the code snippet below, I present an example of using **OptionLab** to calculate and display the probability of profit for this strategy. The data used in the calculation refers to Nvidia stock (ticker: NVDA) on January 16, 2023. The maturity of the call option was February 17, 2023. The covered call setup consists of buying 100 Nvidia shares at \$168.99 and selling the same number of \$185.00 strike calls for \$4.10 each.

```
# Import run_strategy()
from optionlab import run_strategy

# Input data (Nvidia, NVDA)
stock_price = 168.99
volatility = 0.483
interest_rate = 0.045
min_stock = stock_price - 100.0
max_stock = stock_price + 100.0
start_date = "2023-01-16"
target_date = "2023-02-17"

# Strategy definition
strategy = [
    {"type": "stock", "n": 100, "action": "buy"},
```

```

    {
        "type": "call",
        "strike": 185.0,
        "premium": 4.1,
        "n": 100,
        "action": "sell",
    },
]

# Run a calculation
out = run_strategy(
    {
        "stock_price": stock_price,
        "volatility": volatility,
        "start_date": start_date,
        "target_date": target_date,
        "interest_rate": interest_rate,
        "min_stock": min_stock,
        "max_stock": max_stock,
        "strategy": strategy,
    }
)

# Print the output
print(out)

```

The input data is passed as a Python dictionary to the `run_strategy()` function, imported from **OptionLab**. This function performs the calculations and generates a series of outputs for the covered call strategy, as shown below.

```

Probability of profit: 0.5472008423945267
Profit ranges: [(164.9, inf)]
Per leg cost: [-16899.0, 409.99999999999994]
Strategy cost: -16489.0
Minimum return in the domain: -9590.0000000000002
Maximum return in the domain: 2011.0
Implied volatility: [0.0, 0.456]
In the money probability: [1.0, 0.256866624586934]
Delta: [1.0, -0.30713817729665704]
Gamma: [0.0, 0.013948977387090415]
Theta: [0.0, 0.19283555235589467]
Vega: [0.0, 0.1832408146218486]
Rho: [0.0, -0.04506390742751745]

```

OptionLab is provided as is, with no guarantee that its results are accurate. As such, I am not responsible for any losses caused by the use of the code. Bugs can be reported as issues on the [project's GitHub page](#).

Options are very risky derivatives and, like any other type of financial vehicle, trading options requires due diligence. I am not a financial advisor and the content of this article is not a recommendation.

I am open to hearing any questions, corrections, comments or suggestions. You can also reach me on [Linkedin](#) or [X](#).

Options Strategy

Options Trading

Quantitative Finance

Python Programming



Follow

Written by Roberto Gomes, PhD

903 Followers · 14 Following

I am a Professor of Materials Engineering specialized in computational research in materials science, physics, and engineering. Also interested in finance.

Open in app ↗

Sign up

Sign in

Medium

Search



What are your thoughts?

[Respond](#)**Dolly Bonde**

May 23, 2024 (edited)



Hey I am using latest version. But I am unable to calculate Probability of profit and greeks on the expiry day and days to expiry becomes 0. Is there any workaround?

BTW fantastic job on the library. Its very useful.



9



3 replies

[Reply](#)**kumar**

Apr 11, 2024



It was interesting article. Looking forward to see more such.



11



2 replies

[Reply](#)**Rainer Mokros**

Feb 5, 2024



Question to the .getdata() the minstock and maxstock are the 52 low and high? or I'm on the wrong side of the road?



14



2 replies

[Reply](#)[See all responses](#)

More from Roberto Gomes, PhD



 Roberto Gomes, PhD

Is Delta the same as In-The-Money probability?

Using OptionLab calculations to address this question

Dec 13, 2023  62  2



 Roberto Gomes, PhD

Python for Options Trading (3): A Trade with 100% Probability of Profit

NOTE: The library has undergone significant changes, rendering the code from the previous version of this article incompatible. As a...

Oct 30, 2023 🖱 590 💬 29

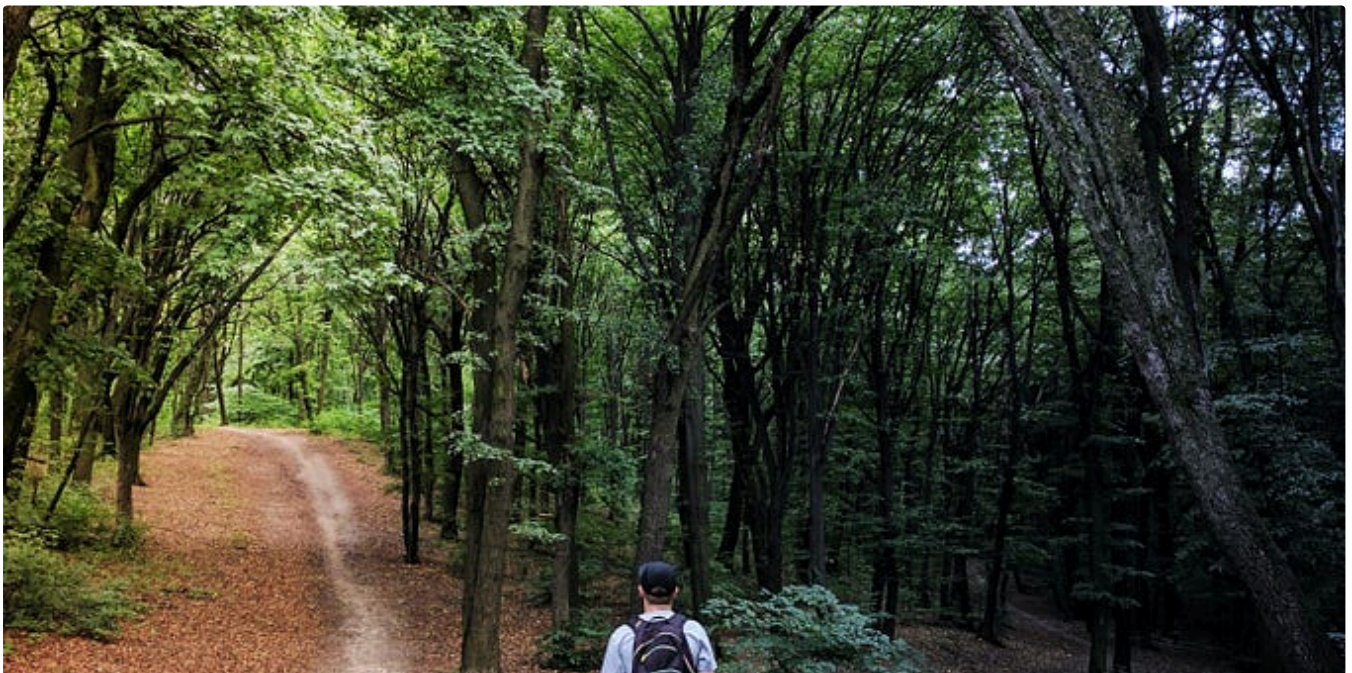


Roberto Gomes, PhD

Probability of profit of an options strategy from the Black-Scholes model

Very important in the evaluation of an options strategy, but often neglected or poorly determined, the Probability of Profit (PoP) is the...

Jul 24, 2023 🖱 29 💬 1



Roberto Gomes, PhD

Python for Options Trading (1): Selecting a Call Spread with the Highest Probability of Profit

EDIT: The library has undergone significant changes, rendering the code from the previous version of this article incompatible. As a...

Aug 7, 2023 🖱️ 93 💬 8



See all from Roberto Gomes, PhD

Recommended from Medium



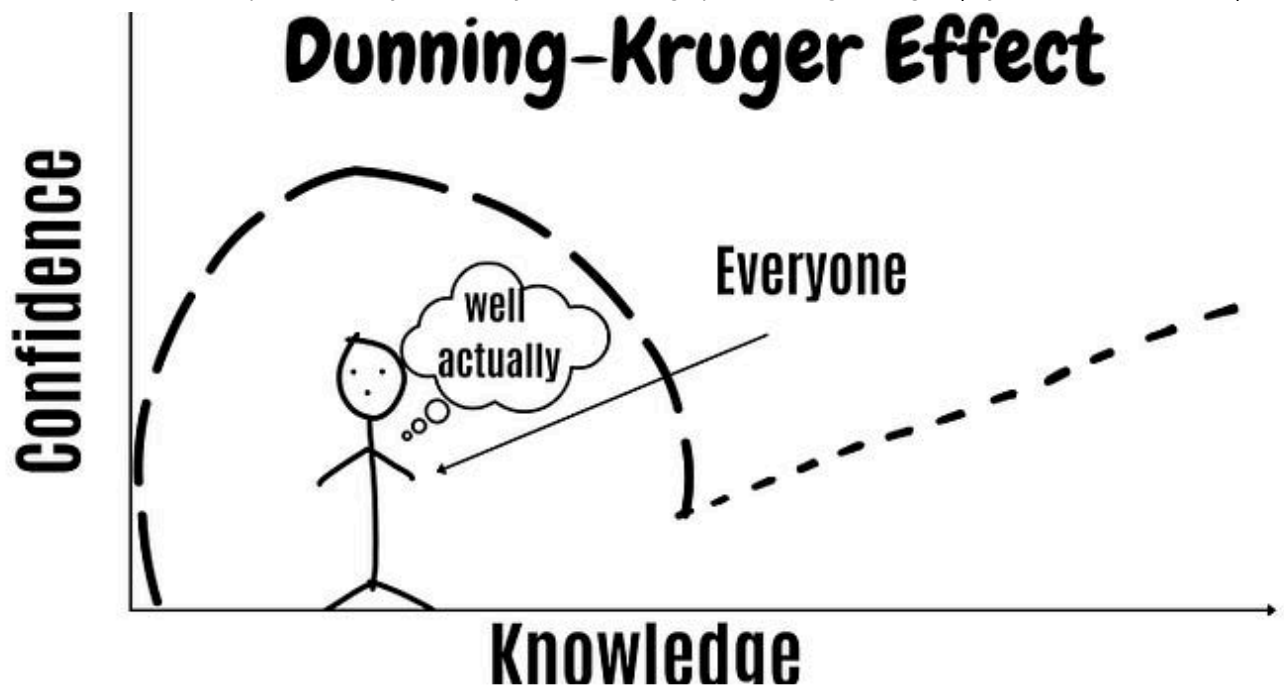
 Unicorn Day

A Simple Yet Powerful Trading Strategy: The Moving Average Slope Method

Ever wondered how the pros know when to enter and exit the market? What if I told you there's a way to ride trends with mathematical...

★ 5d ago 🖱️ 225 💬 3





 In Making of a Millionaire by Ben Le Fort 

Confidently Wrong: How the Dunning-Kruger Effect Impacts Your Money

“The first rule of the Dunning-Kruger club is you don’t know you’re a member of the Dunning-Kruger club”

★ Mar 15, 2023 🖱 2.4K 💬 47



Lists



Coding & Development

11 stories · 1001 saves



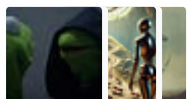
Stories to Help You Grow as a Software Developer

19 stories · 1591 saves



Generative AI Recommended Reading

52 stories · 1647 saves



Natural Language Processing

1928 stories · 1582 saves



In InsiderFinance Wire by Huzaifa Zahoor

How to calculate Bollinger Bands in Python

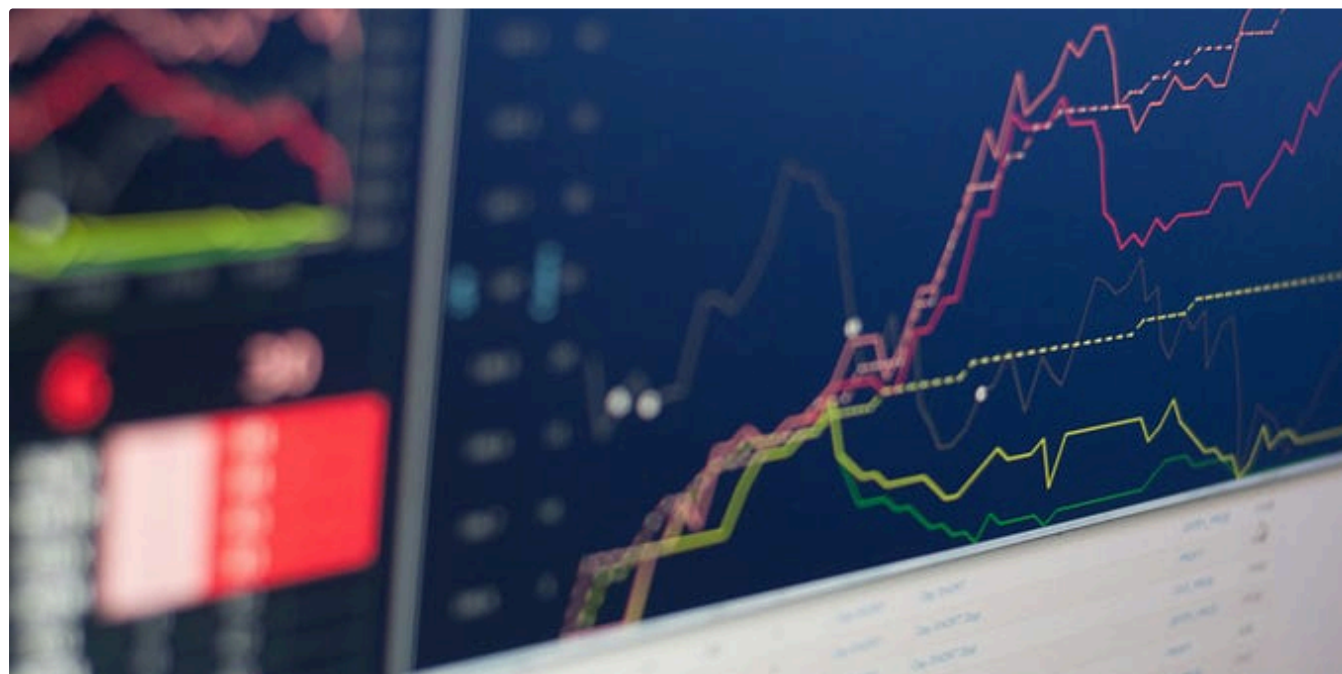
Ready to explore another powerful technical indicator? Let's dive into Bollinger Bands and learn how to calculate them using Python. This...



Aug 26, 2024




119



In Investor's Handbook by Anil Jangra, MBA

How Selling Put Options Transformed My Portfolio

Discover how selling put options on solid dividend stocks like Coca-Cola boosted my passive income. A simple strategy for smarter returns!

Jan 21  175  8 Cristian Velasquez

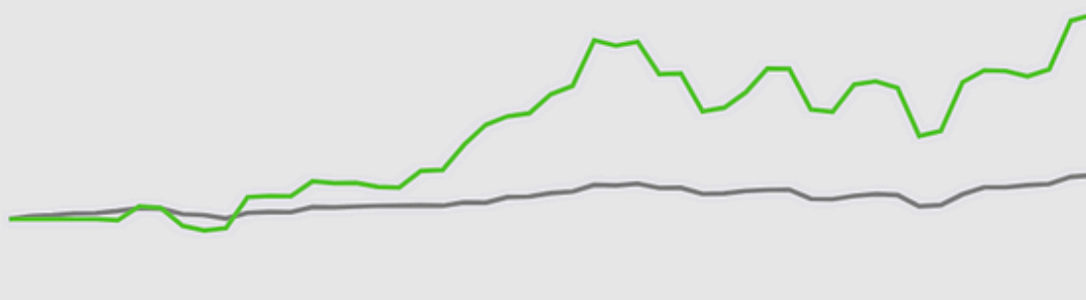
Can YOLO Predict Stock Market Ups and Downs?

The Application of Computer Vision for Predicting the Market Ups and Downs in Real-Time.
End-to-end Solution Included.

★ 6d ago  100  1

(Strategies) Buy 20 percent of buying power in TQQQ Stock when (20 Day TQQQ SMA > 50 Day TQQQ SMA) and (# of Days Since the Last Filled Buy Order of TQQQ ≥ 1)

Click to see more details and adjust backtest settings

 In DataDrivenInvestor by Austin Starks

I used OpenAI's brand new O3-mini model to create a trading strategy. It's DESTROYING the market

You can copy this strategy for yourself in a single click

★ Feb 1 🖱️ 323 💬 13



See more recommendations