## Group A: Pedro Vincenty, Cece Liu, Hannah Bao

Advanced Python for Financial Modeling

Professor Groner

5/12/2023

Options Application

1. **Statement of Problem**

An options contract is a complex financial instrument that gives the rights to investors to buy or sell an underlying asset at a certain price on a specific expiration date. Options contract is also a derivative product that requires some expertise to calculate the profit or loss when adding it to the investment portfolio. Knowing the estimate of the fair value of an option, investors could adjust their trading strategies and portfolios. However, many trading platforms and financial news providers give limited or unexplicit information about each past contract. In addition, for investors who are looking for advice on investing in options, the platforms and news providers do not provide the corresponding information. Therefore, in our project, we want to create an application for investors so that they can retrieve a visual representation of a single-option strategy.

1. **Approach of Problem**

In this section, we will discuss our approach to solving the problem stated in the previous part. Figure 1 shows a general system design of the creation for the application.

In order to provide a concise and clear result for the users, we first need to understand some basic concepts about options and explore different Python packages that best tailor the needs of the user. After doing research, we found the “yahoo\_fin” package that supports us in retrieving relevant options information and “opstrat” package that allows us to graph the payoff diagrams for various strategies. Once we decided on the tools and general idea, we wrote a Python script to retrieve stock information, calculate option prices, and display the corresponding visualizations. Furthermore, we tested the code in the terminal to see whether Streamlit is feasible. Last but not least, when users open the webpage using the terminal, they can input the stock ticker, select strike price and expiration date, and get a profit/loss table and payoff diagrams with three strategies as a result.

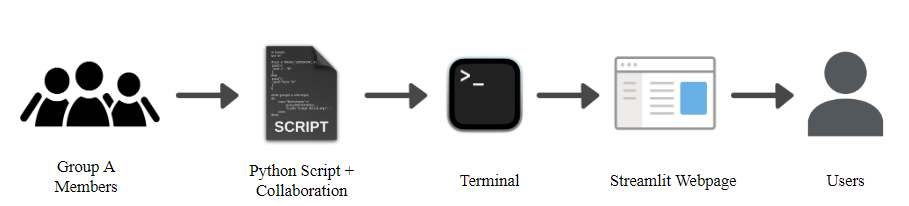


Figure 1: Web Application Architecture

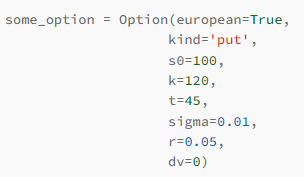
The scope of our project is to provide suggestions about an option on the stock at its expiration date.

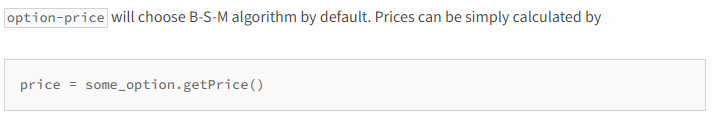
1. **The Models**
   1. **Option Pricing Model: the Black-Scholes Model**

Option pricing models are mathematical equations that estimate the theoretical value of an option using certain variables. The models are powerful for investors who are interested in options trading. There are three popular models: the binomial option pricing model, the Black-Scholes model, and the Monte-Carlo simulation.

Aligning with the scope of our project, we selected the Black-Scholes model as the final option pricing model to calculate the fair value of European-style options. The model requires 6 variables: the spot price of the stock, the strike price of the stock, implied volatility, expiration date, interest rate, and dividend yield.

The Python package we used is called “option-pricing.” First, we need to assign a new variable to the option with all the inputs. Then, we can calculate the option price by specifying the model. An example of the function used is shown in Figure 2.



Figure 2: Example of Using an “option-price” package

One limitation of the Black-Scholes model is that it only applies to European-style options, but it is not applicable to American-style options due to their feature to be exercised before the maturity date.

* 1. **Streamlit Interactive Model**

Streamlit is a framework for creating interactive web applications in Python. Our interactive model built with Streamlit allows the users of our application to interact with the program and Black-Scholes model in real time by inputting data and observing the output (payoff graphs and profit/loss tables). Using Python to create the Black-Shcoles model, we decided to use Streamlit model to create our interactive application for its ease of use and interactive features.

Our Streamlit model imports several libraries that are necessary for the program to run on Streamlit, including Yahoo\_fin, pandas, and streamlit, then uses them to retrieve data about options of interest. Users can enter a stock ticker and choose an options trade date and strike price, then the most appropriate treasury rate is obtained based on the number of days until the option expires. After gaining all the necessary information, the model displays the cash payouts for buying one contract of 100 shares for both a call and put options. The model will also display the range of probable equity prices at expiry, giving a better idea of the potential profit or loss.

1. **Data**

The data we collected to be used in the application is via a Web Service API, called “yahoo\_fin.” The Python package can scrape data from Yahoo Finance website. The types of data available are not limited to stock prices, option prices, financial statements, etc. In our project, we will use the information from option contracts and the current price of a stock. Within the option contracts, there are strike prices, expiration dates, number of contracts existing, implied volatility, etc.

Since we focus on European-style options using the Black-Scholes model to calculate options prices, it is better to find the risk-free rate that is closest to the expiry date than using the 10-year treasury yields. In this case, we attain the risk-free rate by scraping the yield table from the US Treasury website.

1. **Implementation**
2. **Preparation**

Before using the application, users need to first install the necessary Python packages in the Python environment or the system. As streamlit is our main platform for the interaction, users need to type the command “pip isntall streamlit” in their terminal. The installation steps for different computer systems can be found on the [streamlit documentation page](https://docs.streamlit.io/library/get-started/installation). To install the rest of the modules, simply copy and paste the following commands in the terminal one at a time:

pip install yahoo\_fin

pip install opstrat

pip install option-price

1. **Opening the GroupA\_OptionApplication.py file (streamlit)**

After installing all the necessary modules in the environment, type in the command “cd” to change the directory to where the downloaded Python file is located. To activate the Streamlit app, type in “streamlit run GroupA\_OptionApplication.py” to run the code, shown in Figure 3. The command will direct the user to the Streamlit application in the browser.



Figure 3: step to open the Streamlit app in your browser

1. **Inputs**

Once the webpage is opened, the application will prompt the user to enter a stock ticker of interest. Then, the program will scrape the expiry dates and strike prices of the available option contracts from Yahoo Finance using the yahoo\_fin so that the user could choose correspondingly. Moreover, user can determine the trading strategies of their interest, long or short. Figure 4 demonstrates an example of AAPL stock.

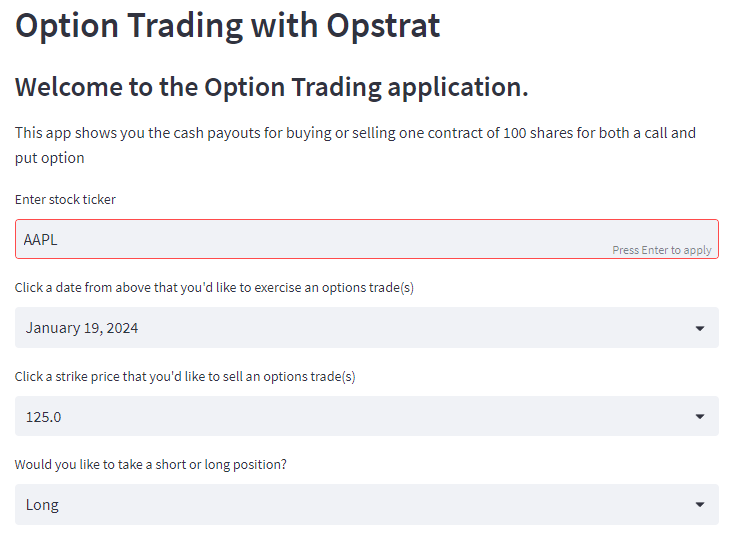


Figure 4: Input of an option on AAPL

1. **Outputs**

As soon as the user has inputted the stock, expiry date, and strike price, the Streamlit app will generate several profit/loss tables and corresponding graphs. We want to let users see both possibilities (call and put options) in their indicated position. In the end, an example shows one possibility of a strangle on the specified stock. A strangle is “a simultaneous purchase of options to buy and to sell a security or commodity at a fixed price, allowing the purchaser to make a profit whether the price of the security or commodity goes up or down. [3]” The graphs are formed by using the “opstrat” Python package. Figures 5-7 illustrate the example of visualizations for AAPL stock.

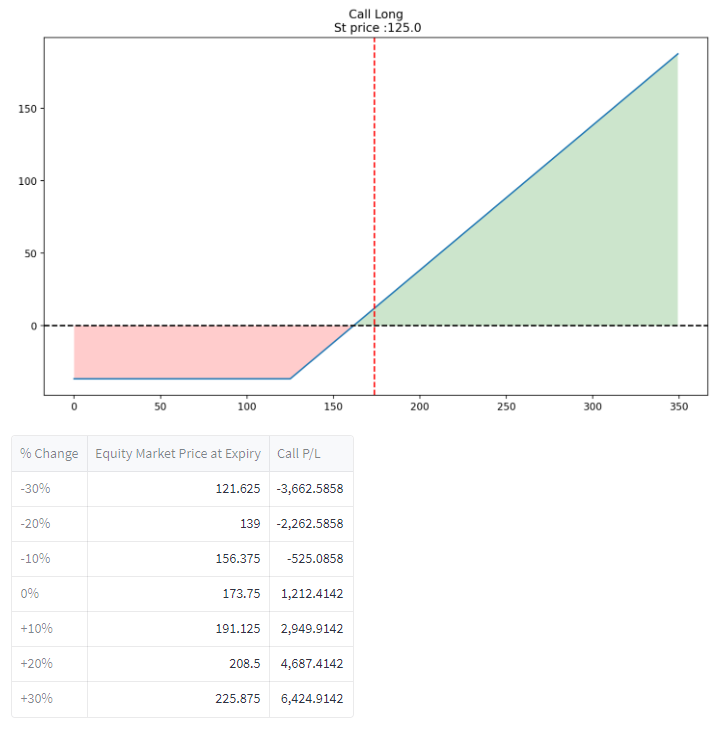
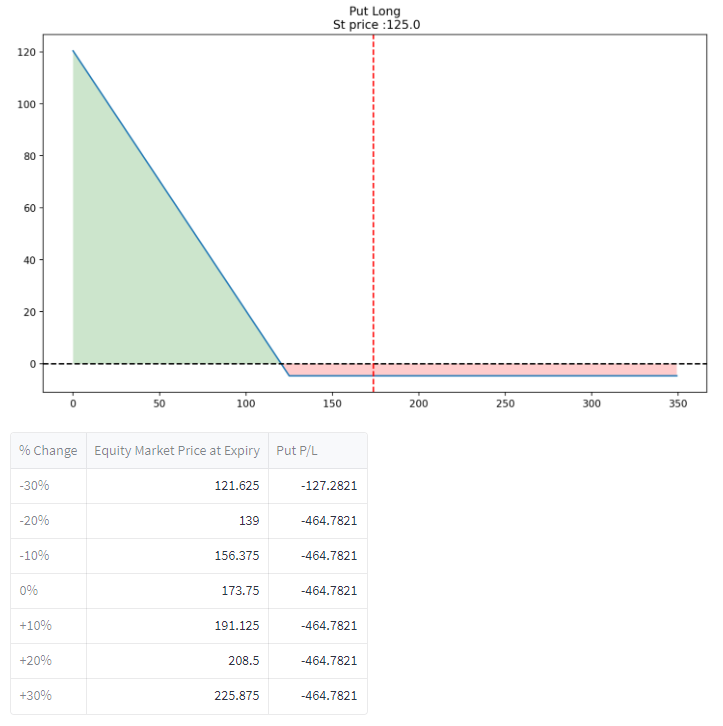


Figure 5: Long Call Plot & its P/L Table

Figure 6: Long Put Plot & P/L Table

A screenshot of a graph

Description automatically generated with medium confidence

Figure 7: 2 Options Plot & Total P/L table

1. **Results and Conclusions**
   1. **Results**

We have successfully developed a user-interactive interface that helps investors gain a better understanding of the potential gain or loss associated with various options based on their inputs. This is particularly important given that options contracts are complex financial instruments that require specialized knowledge to calculate profit and loss, and many trading platforms and financial news providers do not provide adequate information about past contracts. Our application aims to fill this gap by providing a visual representation of single-option strategies, allowing investors to adjust their trading strategies and portfolios accordingly. With our application, investors can more easily navigate the complex world of options trading, and make informed decisions based on a clear understanding of the risks and rewards involved.

* 1. **Future scope**

In the project, we created an application for two types of options and listed an example of a strategy for users’ interests. Nevertheless, the combination of option trading strategies has many variations. For instance, investors may want to long call options at different strike prices, or they want to buy and sell different types of options. There are many interesting and in-depth analysis one can apply based on our application and upgrade it to attract more users.

1. **Lessons Learned**
   1. **Became more familiar with the concepts of options**

By creating a Streamlit application on options, we were able to gain a deeper understanding of the underlying concepts of options trading, including the concepts of the strike price, expiry date, call and put, long and short, etc. The interactive nature of the application allowed us to experiment with different scenarios and see how the changes in those key variables could affect the value and potential profit or loss of an option/options portfolio. Additionally, by building the application ourselves, we were able to get a more in-depth understanding of the mathematical models and calculations (Black-Scholes, Monte-Calo simulation, etc.) used in option pricing and valuation, leading to a more comprehensive understanding of this complex financial instrument.

* 1. **Learned new packages and streamlit app**

To simplify our codes and make our application more adaptive to different scenarios (call and put, different expiry dates, and different strike prices), we researched and noticed that some specialized packages such as Yahoo\_fin and opstrat are designed specifically for obtaining information from Yahoo Finance and calculating the payoff graphs for options. With these packages, we can simply use several lines of code to conduct complex Black-Scholes model calculations for the profit and loss of the option or the options portfolio at each point.

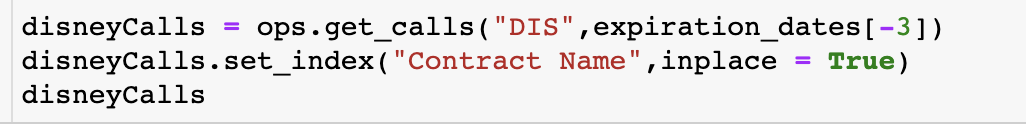


Figure 7: yahoo\_fin method get\_calls to get the information of call options for disney

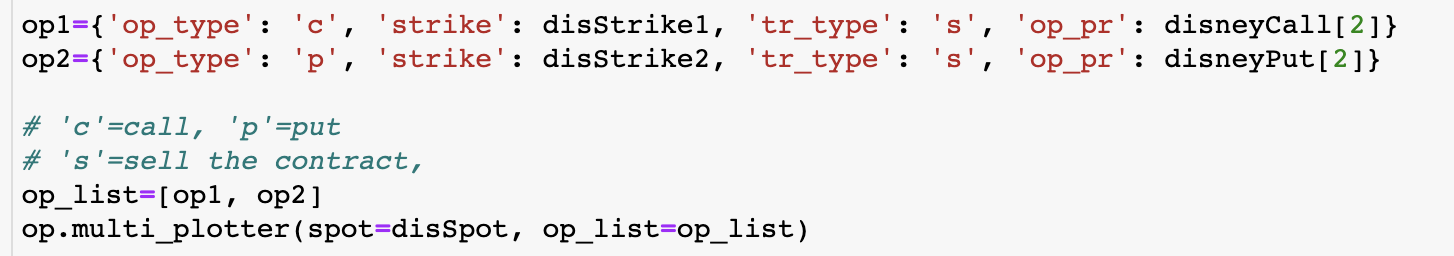


Figure 8: opstrat method multi\_plotter to plot the payoff graph for multiple options

1. **References**
2. <https://sanketkarve.net/automating-option-pricing-calculations/>
3. <https://corporatefinanceinstitute.com/resources/derivatives/option-pricing-models>
4. <https://docs.streamlit.io/library/get-started/installation>
5. <https://pypi.org/project/opstrat/>