Options Forecasting

Steven Miller

Abstract

This paper will look at the domain of financial derivatives, particularly stock options. By examining a dataset of historical option pricing and relevant data, potential trades will be analyzed to identify criteria that can be used to project the potential profitability of the trades.

Options trading is similar to stock trading, but with added dimensionality. Option contracts are priced based on a "strike" price and an expiration date in relation to an underlying stock symbol. There are two types of option contracts, "calls" and "puts".

A call contract gives the owner the right to purchase 100 shares of a stock for a given strike price, regardless of the price of the stock at the time. For example, one could purchase a call contract for KO (Coca-Cola) with a strike price of $60 and an expiration date one month into the future. A premium would be paid for this contract, perhaps of about $1 per share depending on numerous factors. If the price of KO were to go up to $65 before the contract expires, the owner of the contract could exercise the contract and purchase 100 shares for $60. The net profit from this trade would theoretically be the market value of the stock ($65 per share times 100 shares), minus the cost of the contract ($1 per share times 100 shares) and the cost of purchasing the stock at the strike price ($60 per share times 100 shares). A net profit of $400 in a one-month time span on a $100 investment.

The other type of contract, a put contract, gives a holder the right to sell 100 shares of a stock at a given strike price, regardless of the price of the stock at the time. To keep the example similar, a $60 strike price put contract in KO with a 1-month expiration would allow the holder of the contract to sell 100 shares of KO stock for $60 per share, even if the stock had fallen to $55 per share in that time.

In addition to buying these contracts, they can also be sold. This puts the trader on the other side of each scenario, essentially hoping that the stock remains below $60 per share in the call scenario, or above $60 for the put scenario. In these cases, the seller of the contract would collect the premium for the contract,

and the contract would then expire worthless. If the contract did have value at expiration, the seller would be liable for the difference between the current share price and the strike price of the contract. In the case of "black swan" events such as the financial crisis of 2008, this could result in significant losses beyond what a stockholder would have experienced.

The Data

The dataset in use is a large historical dataset of every option contract traded since 2002. The data contains information on stock prices, as well as the pricing of options for every trading day in the time frame. The option data includes some other information, such as the *implied volatility* of the stock (how much the stock is expected to move based on the market price of the options), and several *Greek* values:

- Delta – how much the price of an option is impacted by the change in the stock price (can be between -1 and 1). A delta of 0.5 would indicate that an option would increase in price by 50 cents for every dollar the stock price increases.
- Gamma – the first derivative of delta, a measure of the change in delta of an option
- Theta – how time impacts the price of an option. The rate of decay in the value of an option each day. A theta of 3 would indicate that an option would decrease in value by 3 dollars in a day due to the passage of time.
- Vega – how volatility impacts the price of an option. A vega of 0.25 would indicate a 25 cent increase in price for every percentage point that implied volatility increases.

The Process

There are numerous option trading strategies that can be created by combining the buying and selling of several contracts at the same time. For the sake of simplicity, those will be ignored for this project, though I would like to incorporate them in the future. The focus of this project will look at the potential

profitability of selling put contracts. If done correctly, this should result in a strategy that has a high probability of being consistently profitable but also runs the risk of large losses if not properly managed.

The first step was to take the dataset from its raw form as a collection of 330 gigabytes of CSV files and load everything into a PostgreSQL database to make it easier to work with. A Python script to read each file and perform some minor transformations such as parsing dates was created to assist with this process.

The next step was to create an initial OLS regression to get an understanding of the relationships between the Greek values and implied volatility to the eventual profitability of a trade. The initial results were relatively poor, with an adjusted R^2 value of 0.07, but I chose to go forward with the model in the short term, as profitability can vary wildly and events that can lead to losses would be difficult to predict.

With the process now in place, I wanted to create a benchmark to compare the eventual results against. I decided the best comparison would be to look at SPY, a fund that matches the S&P 500 index, and compare a strategy of selling put contracts to a strategy of buying the fund and holding it through the same time period. A simulation of $10,000 invested in SPY shares at the start of 2006 was found to be worth $34,316 at the end of 2019.

Next, I focused on creating a full pipeline for testing. The first step was selecting 10,000 trades at random that occurred before the date being tested. These trades were transformed to be fed into the regression model. This model then analyzed all trades available on a given day and predicted the profitability of each trade. The trade identified as the most profitable was selected and the trade simulated. When this trade was closed, the backtesting system moved on to the next trade using the same process until the end of 2019 was reached. The result was a portfolio value of $18,543, not as good as the benchmark strategy, but profitable.

Building upon the OLS regression, an XGBoost regression model was implemented next. Using an early stopping method to avoid overfitting, the model was tested through the same cycle of 2006 to 2019 with a

final result of $23,789. This was an improvement over the OLS regression, but I believe there is room for further improvement yet as it's still well short of the benchmark value. On challenge of the XGBoost model was that a stark difference existed between the performance on training data and the performance on validation data. I believe this may be a result of overfitting and will be looking to adjust the hyperparameters to reduce this. Part of the problem is that hyperparameters were tuned using random splits of data to validate. This is generally fine, but in the past I have found that with some time series data, this can lead to overfitting even on the validation set. This is because data has been split randomly, and the training set, for example, may have Monday, Wednesday, and Friday of a given week, while the test set has Tuesday and Thursday. The reality often is that not much is changing from day to day, and a model that is trained to predict Monday and Wednesday can fill in the gaps and predict Tuesday quite well. I'll be looking to re-tune these parameters using data split in a more blind fashion.

Overall, I would consider this project to be a success. I also feel there is a large amount of room for improvement yet with the strategy. One current bottleneck is the time it takes to perform a backtest using simulated trades. Currently, a system is in place that uses 10,000 contracts randomly selected. These contracts are then used to simulate 10,000 trades, the results of which are used to train the model. This is a scenario where more data would be better, but it currently takes approximately two minutes to simulate 10,000 trades, making backtesting a 14-year time frame rather unfeasible. My current belief is that if I simulate every trade in advance and store the results in a database table, the time to retrieve these results would be lower than the current time to simulate 10,000 trades and allow for faster testing with more data available for use.

I'd also like to build upon what currently exists. While selling puts is a great start, and the strategy I use most often personally, it's one of four basic strategies that could be implemented. Adding the other three would give the pipeline more choices for selecting a profitable strategy on each date. More choices should only lead to better results in this situation. In addition to these four basic strategies, there are a number of

more complex strategies available that are essentially various combinations of the four basic strategies. These include strategies such as strangles, which involves selling a call and a put option at the same time, or vertical spreads which involve limiting the overall risk of selling a contract at the expense of a lower potential profit. These would all be worth testing and implementing, but there hasn't been enough time to do so yet. I look forward to building upon this.

Currently, for training purposes, I am using any data that would have existed prior to a day being entered. I'm concerned that some data may be too old to be of particular usefulness at the time of a trade. I'm not sure how relevant a trade from 2002 is when trying to predict the performance of a trade in 2019. I'm considering two options for reducing the impact of this. One would be adding the age of data as a feature. I'm not sure how successful this would be. The alternative is to begin limiting the pool of training data to a certain window, say the past three years of data. I believe this is the better option and will experiment with finding a number that works best. While I want to remain relevant, I also want to have a pool of data that captures various market conditions successfully.

Finally, moving forward, I would like to take what currently exists and build onto it from a single pipeline to test a single strategy to one that can analyze an existing portfolio and recommend a new trade based upon what already exists in the portfolio. By identifying stock tickers which have minimal correlation with existing tickers in a portfolio, trades could be added in a way that helps to intelligently diversify.

I'm satisfied with the results of this project so far. The performance isn't quite up to the benchmark level yet, but there's a lot left on the table. Plenty more can be tested and simulated, and I believe the gap will be closed as additional strategies are added and models are tweaked. This is a decent start, but there's plenty to add yet, things to tweak and optimizations to be made. I look forward to working on this for quite a while.

References

Anselmo, P., Hovsepian, K., Ulibarri, C., & Kozloski, M. (2008). Automated Options Trading Using Machine Learning. https://pdfs.semanticscholar.org/6f27/d1928195208d1fb2d6457bbd2950cfa289d4.pdf

Bain, A., Tiwaree, P., & Okamoto, K. (n.d.). Recognizing Informed Option Trading. Retrieved February 9, 2020, from http://cs229.stanford.edu/proj2009/BainOkamotoTiwaree.pdf

Craig, T. (2018, July 12). Options Theory: Short Puts are a Pro's Long Stock | Tackle Trading. Tackle Trading. https://tackletrading.com/options-theory-short-puts-pros-long-stock/

Divakar, V. (2018, January 17). Covered Call Strategy using Machine Learning. QuantInsti; QuantInsti. https://blog.quantinsti.com/covered-call-strategy-machine-learning/

Hubert, A. (2019). Using Deep Learning for Better Option Pricing. Dataiku.Com. https://blog.dataiku.com/using-deep-learning-for-better-option-pricing

Investopedia. (2019). Options Basics Tutorial. Investopedia. https://www.investopedia.com/options-basics-tutorial-4583012

Kreimer, A. (2018, May 19). Things You Learn After 1 Year of Day Trading for a Living. Medium; Noteworthy - The Journal Blog. https://blog.usejournal.com/things-you-learn-after-1-year-of-day-trading-for-a-living-a97bbc8d19fa

Mezofi, B., & Szabo, K. (2018, November 23). Beyond Black-Scholes: A New Option for Options Pricing. We Are WorldQuant. https://www.weareworldquant.com/en/thought-leadership/beyond-black-scholes-a-new-option-for-options-pricing/

Sunith Shetty. (2018, April 14). How to build an options trading web app using Q-learning | Packt Hub. Packt Hub. https://hub.packtpub.com/how-to-build-an-options-trading-web-app-using-q-learning/

TastyTrade. (2015, April). Supervised Learning Models. Tastytrade.Com - A Real Financial Network. https://www.tastytrade.com/tt/shows/the-skinny-on-options-data-science/episodes/supervised-learning-models-04-01-2015