Monte Carlo simulation weighted with sentiment scores from news data to determine call and put option prices

Gourab Hazra

Monte Carlo simulation weighted with sentiment scores from news data to determine call and put option prices

Thesis submitted to the
Xavier University Bhubaneswar
for award of the partial fulfillment of the degree

of

Master of Technology in Data Science and Analytics

by

Gourab Hazra

Under the guidance of

Prof. Rudra Mohan Tripathy

School of Computer Science and Engineering
Xavier University Bhubaneswar
Bhubaneswar - 752 050, India
April 2021

CERTIFICATE

Date:10/04/2021

This is to certify that the thesis entitled **Monte Carlo simulation weighted with sentiment scores from news data to determine call and put option prices**, submitted by **Gourab Hazra** to Xavier University Bhubaneswar, is a record of bona fide research work under my supervision and I consider it worthy of consideration for the award of the degree of **Master of Technology in Data Science and Analytics** of the Institute.

Prof. Rudra Mohan Tripathy
Supervisor
Associate Professor
School of Computer Science and Engineering
Xavier University Bhubaneswar
Bhubaneswar - 752 050, India

DECLARATION

I certify that

a. The work contained in the thesis is original and has been done by myself under the general supervision of my supervisor.

b. The work has not been submitted to any other Institute for any degree or diploma.

c. I have followed the guidelines provided by the Institute in writing the thesis.

d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.

f. Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Gourab Hazra

Dedicated to my family

ACKNOWLEDGMENT

I would first like to thank my supervisor, Prof. Rudra Mohan Tripaty, whose insight in regard to the approach to the problem statement was helpful. For generating an interest and giving in depth explanations of the workings of important concepts in the field of Deep Learning. I would also like to thank my professors, Prof. Rakesh Prasad Badoni and Prof. Chandan Misra, for their critically helpful and valuable advice throughout my studies. I thank profusely Prof. Rajiv Bhutani for teaching the course of Financial Analytics triggering my curiosity for the subject and its possible interaction with data science.

In addition, I would like to thank my parents for their wise counsel and sympathetic ear. You are always there for me.

Gourab Hazra

Abstract

Investor sentiment affects the ups and downs of a stock according to many previous studies. In this study I produce an empirical method of pricing stock options using sentiment weights in the way that they are less dependent on rigid measures that are used in models like Black Scholes model. Also they are not as unreliable as a traditional Monte Carlo Simulation model. I introduce sentiment weights into the picture in order to reign in on the randomness created by the Monte Carlo simulation. Which helps create a more consistent yet dynamically obtained call and put option pricing. As a part of previous research, I had also seen the effects of sentiments playing a role in the market value of a stock. The weights generated from that previous study is used herewith to base the Monte Carlo Simulation on. I find out that the sentiment inputs provide great potential for option pricing improvements.

Keywords: Sentiment, Options pricing, Black Scholes model, Monte Carlo

Contents

List of Figures

List of Symbols and Abbreviations

List of Abbreviations

ITM	In the Money
OTM	Out of the Money
PCR	Put-Call Ratio
VIX	The Cboe Volatility Index
MC	Monte Carlo
USD	United States Dollar
VAR	Value at risk
CVAR	Conditional value at risk
NLTK	Natural Language Toolkit
LSTM	Long Short Term Memory
CNN	Convolutional Neural Networks
DNN	Deep Neural Networks
S&P	Standard & Poor's
HTML	HyperText Markup Language
VADER	Valence aware Dictionary for Sentiment Reasoning
MPT	Modern portfolio theory

Chapter 1

Introduction

In the long run valuations can drive stock prices, but in the short terms market sentiment become the movers of price. Which is useful to create investment opportunities for long term investors to find attractive entry points and for active traders to both enter and exit positions. Due to the recent years in evolution of deep learning technology and improvement of text analytics algorithms that generate more relevant and informative sentiments. Application of such technology to sentiment analysis, which is empirically one of the most reliable indicators of future price movements is desirable.

Market Sentiment is a qualitative measure of the attitude and mood of investors to financial markets in general. Positive and negative sentiment drive price action, hence creating investment opportunities for active and long-term investors.

Pricing of options has been reliant on quantitative factors like current market price, strike price, volatility, interest rate, and time to expiration to theoretically value an option. Few commonly used models to value options are Black-Scholes, binomial option pricing, and Monte-Carlo simulation.

The main goal of option pricing theory is to calculate the probability that an option will be exercised, or be ITM, at expiration and assign a dollar value to it. The underlying asset price (e.g., a stock price), exercise price, volatility, interest rate, and time

to expiration, which is the number of days between the calculation date and the option's exercise date, are commonly required variables that are used to derive an option's theoretical fair value.

Due to the advent and development of sentiment analysis and Text Analytics in the recent years the possibilities of incorporating the results of these techniques which earlier would be cumbersome to do manually or simply not possible at scale. A new door has opened for us to explore to find out the implications of using these new techniques to find a better suited model that is more dynamic in nature, yet respects the older working mechanisms.

I will also go through the existing studies that have been done related to this niche which are mentioned in the Literature Review section. The approach i have taken relative to the previous studies is unique in the sense that older studies have used existing sentiment score indices. Where i have used an NLTK library that is popularly used for short term trading in many places.

Chapter 2

Problem Statement

Options pricing theory derives various risk factors or sensitivities based on a set of predefined inputs, which are known as an option's "Greeks". Since market conditions are constantly changing, the Greeks give a way to traders with a means of determining how sensitive a specific trade is with respect to price and volatility fluctuations, and the duration of time.

The more the time an investor has available to exercise the option, the greater the likelihood that it will be ITM and profitable at expiration. Marketable options require valuation methods that differ from non-marketable options. Actual traded options prices are determined in the open market and, as is the case with all assets, the value can differ from a calculated theoretical value. However, having the theoretical value allows traders to get an idea of the likelihood of profiting from trading those options.

The Black-Scholes model which is one of the most highly regarded pricing models, assumes stock prices follow a log-normal distribution because asset prices cannot be negative. The model also assumes volatility remains constant over the option's total lifespan. This is unrealistic, and normally not the case, because volatility fluctuates with the level of supply and demand.

Changes to options pricing models include volatility skew, which is a plot of implied

volatilities for options graphed across the range of strike prices for options with the same expiration date. The resultant shape often shows a skew or "smile" where the implied volatility values for options further out of the money (OTM) are higher than for those at the strike price.

Additionally, Black-Scholes assumes that the options being priced are in the European form, executable only at maturity. The model does not take into account the execution of American style options which differ from the European options in the way that they can be exercised at any time before, and including the day of, expiration.

Whereas, the binomial or trinomial models are able to handle both styles of options because they can check for the option's value at every point in time during its lifespan. The key assumption for the binomial model is that there are just two possible results for the stock. The two possible outcomes are either in the up direction or lower i.e. The price will go up, or it will go down. The probabilities with they go up or down are also an assumption. I will need to assume the likelihood of both the stock prices at the end of the binomial iteration. It is not possible to predict which way the price will go with any certainty. All investments carry risks. This is where Monte Carlo comes in.

Chapter 3

Motivation

After going about the different existing option pricing models in the current scenario. There was a realization of the missing out on potential for Deep Learning and sentiment analysis methodologies in the field. Having discussed and realized the drawbacks of existing option pricing models.

The Monte Carlo method used in many places ranging from financial use cases to making hydrogen bombs. It uses a random sampling of information over a set amount of iterations. Given the computation power of modern day machines, the Monte Carlo method can be used in sub-domains like corporate finance, options pricing, and especially portfolio management and personal finance planning.

In certain ways, the Monte Carlo simulation is able to provide best of both the Black-Scholes and binomial worlds. It takes into factor the randomness of the stock price direction, unlike binomial method. It can dynamically generate the values for any time period, which is a drawback in Black-Scholes.

The drawback to the traditional way Monte Carlo is simulated is that it can't account for bearish markets, recessions, or any other kind of financial crisis that might impact potential results. I look to change that by including sentiment weights as initial parameters for my simulation rather than going with randomly assigned portfolio

assignments.

Chapter 4

Existing Literature

Currently existing works show that financial data extracted from digital media in an automated fashion contribute to valuable trading signals. The effect of news specific to a particular company is indeed reflected with the stock price at different intervals - [1].

[2] Here I get to learn that news can bring "noise" into the market. Sentiment cannot be directly observed rather I must rely on a proxy that reflects the true level of investor sentiment. Sentiment has greatest effect on small stocks. Existing metrics like VIX (CBOE Implied Volatility Index). Computed on a real time basis. Baker and Wurgler [BW_sent], University of Michigan Consumer Sentiment Index [UMICH_SENT], American Association of Individual Investors (AAII), weekly Commitment of Traders (COT).

[3] They construct a company sentiment indicator using both content from Dow Jones and Web comparing the results that they obtain. To find that Dow Jones and both web sources contain valuable information that helps predict future stock returns.

[4] Importance of social media has become prevalent in today's exponentially connecting information era. Social media platforms dedicated to stock investments are also on the rise globally. Users are separated into high and low stock-coverage groups and that average sentiment is used to find an appropriate trading strategy.

[5] In this paper the authors are interested to predict the sentiment and aspects of financial micro blog posts and headlines. Every specific company had been assigned a scale that varied between 1 and -1. They had used SVR (Support Vector Regression) to predict the sentiments. Data was acquired from micro blogs and categorized into 3 segments (Positive, Negative, Neutral). Important feature engineering components included N-Grams, Tokenization, Word Replacement, Word Embedding. After conducting their work and calculating accuracy's they reckon the results can be enhanced using LSTMs or simple neural networks.

[6] In this study a Bag of Words model is used along with a finance-specific sentiment lexicon to model the relations between sentiment information and financial risk. Broadly divided into two tasks that constitute regression and ranking, they take into consideration volatility. The finance specific lexicon consists of 6 lists. They use 5-year financial report data. For example, 1996-2000 is taken as training data and the outcomes are tested on the reports of year 2001. To create a word map that associates words with the positive and negative sentiments.

[7] Here they develop a novel method to combine deep learning and classical feature-based model using a Multi-Layer Perceptron for financial sentiment analysis. The model is based on Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). An ensemble is used based on Support Vector Regression (SVR). They get impressive results on a benchmark dataset (SemEval2017). Their work includes qualitative error analysis which correctly identifies wrongly assigned sentiments. They mention about creating an end to end stock prediction system where the system would predict the future stock prices based on the sentiment score and stock value of the company.

[8] In this paper 3 Deep Learning models (DNN, CNN and RNN) were used to perform sentiment analysis. CNN was found to offer the best trade-off between processing time and accuracy. DNN did the worst out of the three.

[9] While rational assessment of information is important, it is stated emotional sentiment also plays an important role in investment decision making. They analyse twitter data about individual S&P 500 firms and compared this to stock returns of those firms. And they find that sentiment in tweets about a specific firm from users with less than 171 followers had significant impact on the stock returns on the next trading day, next 10 days and the next 20 days.

[10] High sentiment predicts low market return and high arbitrage returns. The eigenvalue-decomposition of the mean-variance portfolio is used to show its performance is the sum of market component and arbitrage components. They use shrinking covariance matrices to get optimal mean-variance portfolios.

[11] Here they explore how PCR fares vs VIX to represent variations in assets prices not explained by economic factors. They find that the PCR is a better measure of such factors than VIX and recommend to use that as a deciding factor for choice of market sentiment.

[12] Here they conclude that sentiment affects the cost of capital. Which is why it may have real consequences for the assignment of corporate investment capital between safer and more speculative firms.

[13] Their model shows that the price of call option is multiplied by bullish stock sentiment, and is reduced by stock bearish sentiment, and vice-versa for the put option. They find the price of call option is more sensitive to bullish stock sentiment. The price of put option is more sensitive to bearish stock sentiment. Third, the price of call option increases substantially with respect to the stock sentiment and the option sentiment. And likewise, the price of put option decreases substantially with respect to the stock sentiment.

[14] They find out here their sentiment pricing model which is based on the VIX index, is able to generate option prices that are more consistent with market prices. Although it does not perform as well as for ITM. Overall they find the sentiment pricing

model works better for options with longer maturities.

[15] The author finds that the risk-neutral skewness of monthly index return becomes significantly more negative, and the index option volatility smile steeper, when 1) survey indicates that more market professionals are bearish, 2) large speculators take bigger short positions in S&P 500 index futures, and 3) the S&P 500 index level is more depressed relative to the fundamental. The converse being true likewise.

Chapter 5

Methodology

In this section I explain the various components involved in my study. I will look at each of the components and explain their role in the whole picture. I briefly elaborate as to why I choose each of the methods used in my study.

5.0.1 Beautiful Soup

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It has been used to extract the news data from HTML web-page format of the website finviz.com where the last 7 day news data for Apple, Amazon, Google and Tesla Stocks.

5.0.2 VADER Sentiment Analysis

VADER (Valence aware Dictionary for Sentiment Reasoning) is a model used for text sentiment analysis that is sensitive to both polarity (positive/negative) as well as intensity (strength) of emotion. It is available as a directly callable package as part of the NLTK toolkit.

Its emotion detection aspect relies on a dictionary (lexicon) that maps lexical features to emotion intensities called sentiment scores. The sentiment score can be obtained by summing up the intensity of each word in the text.

Its also understands the context of basic sentences Ex: “did not love” is classified as negative because it contains a negation which otherwise would have been classified as a positive because of the usage of the word “love”. In some of the studies mentioned in my literature review VADER has been used as a text classifier that give sentiment scores.

The compound score is computed by summing the valence scores of each word in the lexicon, adjusted according to the rules, and then normalized to be between -1 (most extreme negative) and +1 (most extreme positive). As explained in the paper, researchers used below normalization.

$$x = \frac{x}{\sqrt{x^2 + \alpha}} \tag{5.1}$$

where x = sum of valence scores of constituent words, and α = Normalization constant (default value is 15)

5.0.3 Sentiment Score

Sentiment score is assigned to sentences that indicate a positive, negative or neutral human emotion. According to the academic paper on VADER, the valence imbued score is measured on a scale of -4 to +4, where -4 stands for the most negative sentiment, +4 for the most positive one. 0 is understandably the value given to a neutral sentence. VADER makes use of certain rules to incorporate the impact of each sub-text on the perceived intensity of sentiment in sentence-level text. These rules are called Heuristics. There are 5 of them. Five Heuristics are explained below:

1. Punctuation, namely the exclamation point (!), increases the magnitude of the intensity without modifying the semantic orientation. For example: “The weather is hot!!!” is more intense than “The weather is hot.”

2. Capitalization, specifically using ALL-CAPS to emphasize a sentiment-relevant word in the presence of other non-capitalized words, increases the magnitude of

the sentiment intensity without affecting the semantic orientation. For example: "The weather is HOT." conveys more intensity than "The weather is hot."

3. Degree modifiers (also called intensifiers, booster words, or degree adverbs) impact sentiment intensity by either increasing or decreasing the intensity. For example: "The weather is extremely hot." is more intense than "The weather is hot.", whereas "The weather is slightly hot." reduces the intensity.

4. Polarity shift due to Conjunctions, the contrasting conjunction "but" signals a shift in sentiment polarity, with the sentiment of the text following the conjunction being dominant. For example: "The weather is hot, but it is bearable." has mixed sentiment, with the latter half dictating the overall rating.

5. Catching Polarity Negation, by examining the contiguous sequence of 3 items preceding a sentiment-laden lexical feature, I catch nearly 90% of cases where negation flips the polarity of the text. For example, a negated sentence would be "The weather isn't really that hot.".

5.0.4 Portfolio Variance & Volatility

Portfolio variance is an indicator of risk, of how the aggregate actual returns of a set of securities constituting a portfolio fluctuate over time. It is calculated using the standard deviations of each security in the portfolio as well as the correlations of each security pair in the portfolio. A lower correlation between securities in a portfolio results in a lower portfolio variance. It defines the risk-axis of efficient frontier in MPT. Portfolio variance is:

$$w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + 2w_1w_2Cov_{1,2} \tag{5.2}$$

Where:

- w_1 = the portfolio weight of the first asset

- w_2 = the portfolio weight of the second asset
- σ_1= the standard deviation of the first asset
- σ_2 = the standard deviation of the second asset
- Cov1,2 = the co-variance of the two assets, which can thus be expressed as $p_{(1,2)}\sigma_1\sigma_2$, where $p_{(1,2)}$ is the correlation coefficient between the two assets

Volatility is a formal measure of a stocks risks. The higher the volatility of a stock, the greater its ups and down swings. Whereas the volatility of a portfolio stocks is a measure of how wildly the total value of all the stocks in the portfolio appreciates or declines. Volatility is usually not observable and usually estimated for variance of returns. There is no hard and fixed mathematical definition for it.

5.0.5 Softmax Optimization Algorithm

The softmax regression is a form of logistic regression usually used in deep learning to prep the weights involved in a neural network. The softmax function normalizes the input value into a vector of values that follow a probability distribution whose total sums up to 1. Which is why I am able to obtain non-binary values.

$$S(y_i) = \frac{e^{y_i}}{\sum_{j=1}^{j} e^{y_j}} \tag{5.3}$$

The softmax of y_i is simply the exponential divided by the sum of exponential of the whole Y vector:

Where S(y_i) is the softmax function of y_i and e is the exponential and j is the no. of columns in the input vector Y. I use softmax to effectively handle negative values of sentiment.

5.0.6 Monte Carlo simulation

Monte Carlo simulations are named after the popular gambling destination in Monaco. As chance and random outcomes are central to the modeling technique, which is why they are directly relevant to gambling. The technique was developed by Stanislaw Ulam. A mathematician who worked on the Manhattan Project.

A Monte Carlo simulation takes the variable that has an uncertainty then assigns it a random value. This process is repeated multiple times while assigning the variable in question with many different values. The greater the number of iterations the better the outcome is. Once the simulation is complete, the results are averaged together to provide an estimate of the value at question. It can be briefly given by the following equation:

$$\frac{\Delta S}{S} = \mu \Delta t + \sigma \epsilon \sqrt{\Delta t} \tag{5.4}$$

where S is The stock Price, Δ S is the change in the stock price, μ is the expected return, σ is the standard deviation of returns, ϵ is the random variable and Δt is the elapsed time period.

The frequencies of different outcomes generated by this simulation will result in a normal distribution. The most likely return being in the middle of the curve. Which implies is an equal chance that the actual return will be higher or lower than that value.

5.0.7 Cholesky Decomposition

Cholesky decomposition allows us to reduces a symmetric matrix into a lower-triangular matrix. When multiplied by the transpose, produces back the original symmetric matrix. In my study it allows me to simulate uncorrelated normal variables and transform them into correlated normal variables. The following formulas are obtained by solving the lower triangular matrix and its transpose. These are the basis of Cholesky Decomposition

Algorithm :

$$L_{i,j} = \frac{1}{L_{j,j}}(A_{i,j} - \sum_{k=0}^{j-1} L_{i,k}L_{j,k}) \tag{5.5}$$

where A is the original matrix, L is a lower triangular matrix with real and positive diagonal entries.

5.0.8 Call and Put Option Pricing

A call option is bought if the trader expects the price of the underlying to rise within a certain time frame whereas, the converse is true for a put option. In the money implies the underlying asset price is higher than the call strike price.Out of the money means the underlying price is lower than the strike price. At the money means the underlying price is equal to the strike price.

The intrinsic values are given by: Max(S-K,0) for calls, Max(K-S,0) for puts. Where K is the strike price and S is the price of asset at maturity.

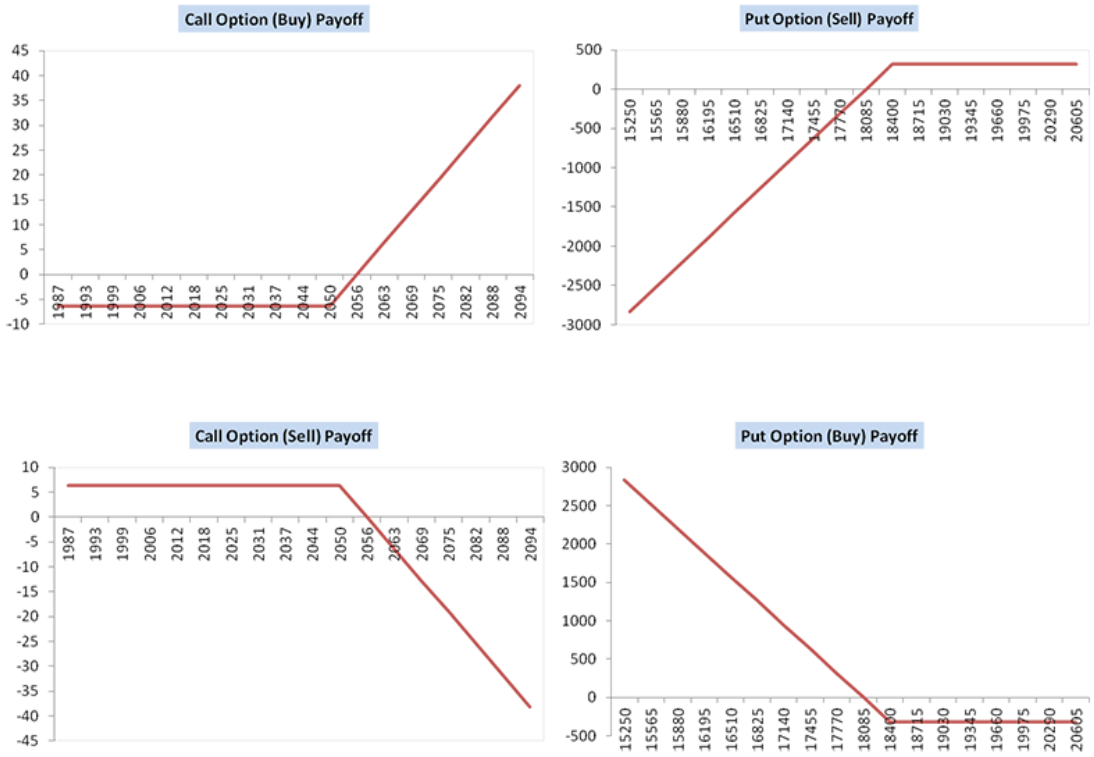

Figure 5.1: Call and put option pricing

Source: `https://zerodha.com/varsity/chapter/summarizing-call-put-options/`

Chapter 6

Results

A sample of the dataset I generate by scraping news data is shown in figure 6.1

	ticker	date	time	headline	neg	neu	pos	compound
0	AMZN	2021-04-10	09:56AM	Amazon Stock Runs Up Towards Resistance: Techn...	0.000	0.841	0.159	0.1779
1	AMZN	2021-04-10	09:33AM	What Amazon Can Learn from a Bruising Victory ...	0.000	0.841	0.159	0.1779
2	AMZN	2021-04-10	09:10AM	Far-Right Extremist Planned to Blow Up Amazon ...	0.000	0.855	0.145	0.1779
3	AMZN	2021-04-10	08:24AM	Chinese regulators slap Alibaba with record $2...	0.000	0.638	0.362	0.3400
4	AMZN	2021-04-10	07:10AM	For its critics, Amazon can never do enough	0.222	0.606	0.172	-0.1280

Figure 6.1: Dataset generated from scraping finviz.com

I can see, there is news data that varies from hours to minutes across the day. There are 3 indicators from the VADER sentiment analysis which are a negative, neutral or positive confidence of the headline. And a compound value which is what is used as weights for the coming application in my Monte Carlo simulation.

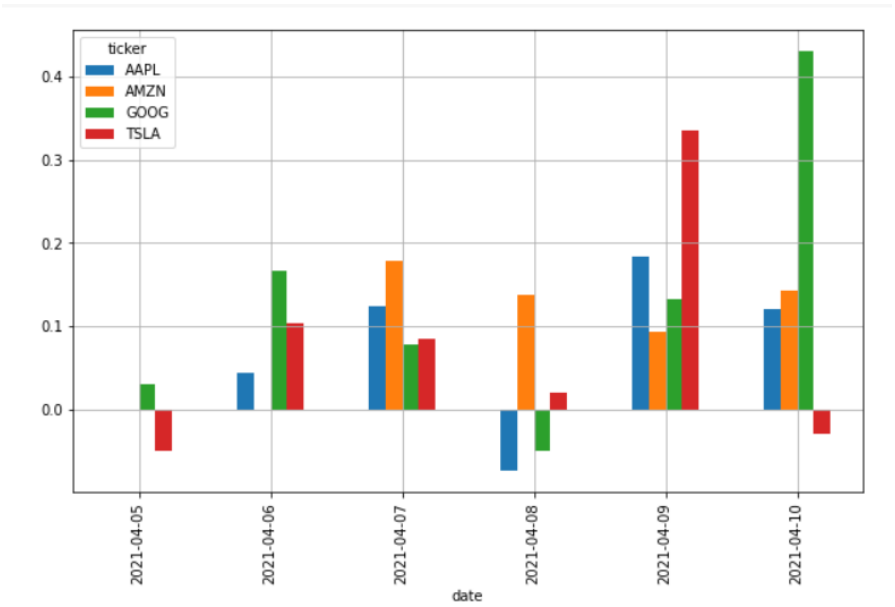

Figure 6.2: Comparison of sentiment of Apple, Amazon, Google and Tesla over a 5 day period

Figure 6.2 Shows a comparison of the various stocks taken into consideration viz. Apple, Amazon, Google and Tesla.

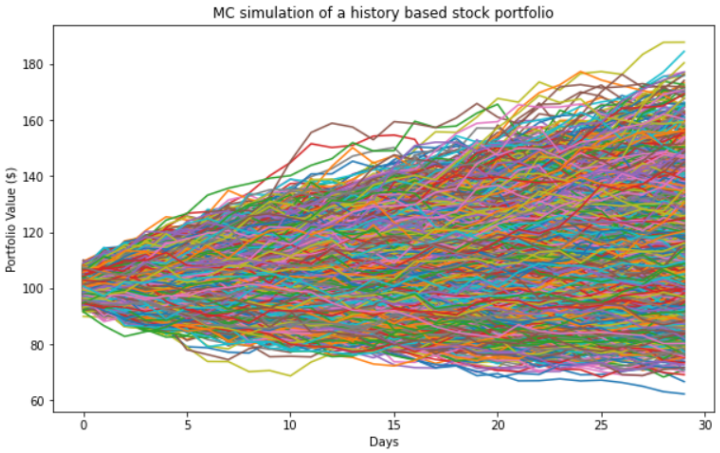

Figure 6.3: Monte Carlo simulation based on random weights

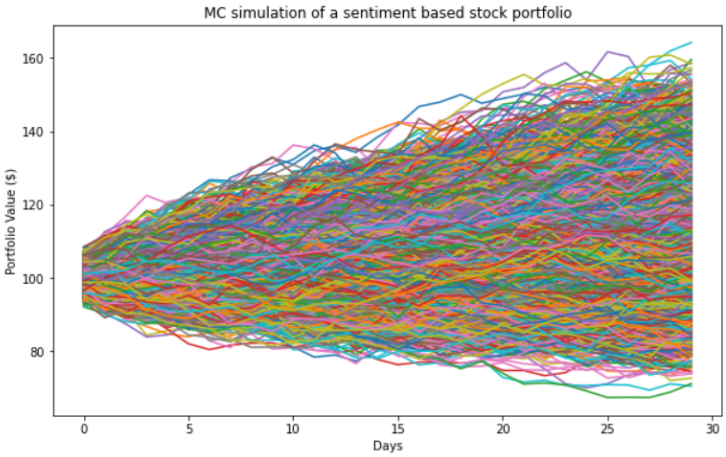

Figure 6.4: Monte Carlo simulation based on sentiment weights

Figure 6.3 and 6.4 show the Monte Carlo simulations generated from random weighted values versus sentiment weighted values over a period of 30 days with 10,000 simulations and a initial strike price of 100 USD.

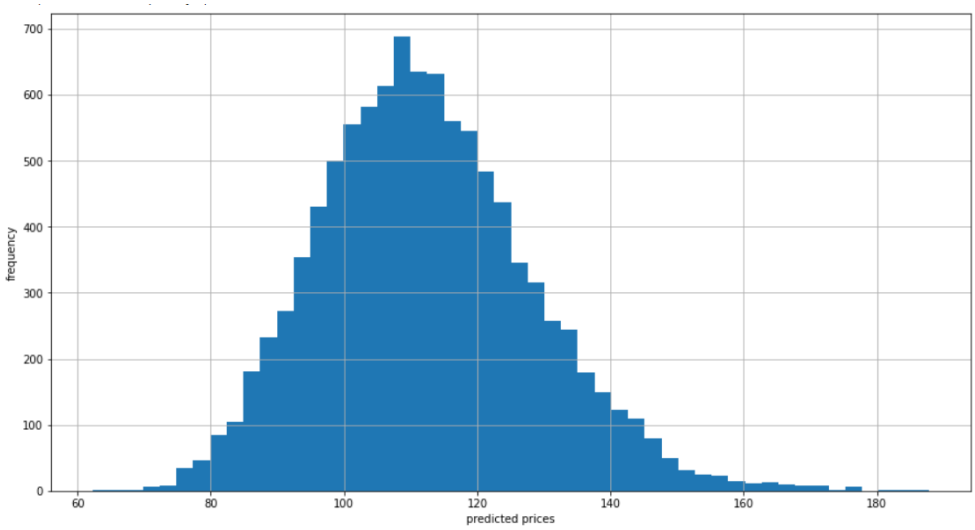

Figure 6.5: Frequency vs. Price plot for the simulation

In Figure 6.5 shows us the frequency vs Price plot. Which on initial glance tell us the expected value of the simulation is somewhere on the lines 110 USD. I then calculate

the exact payoff to get a better idea.

```
Call historical payoff: 5.97885986979855
Call sentiment payoff: 6.741437853698609
Put historical payoff: 1.3707447959171182
 Put sentiment payoff: 1.312583892356049
```

Figure 6.6: Payoff of call and put prices compared between random weights vs. sentiment weights

Here I notice the payoff generated from the sentiment weight is more consistent and does not have the unpredictability of the random weighted simulation.

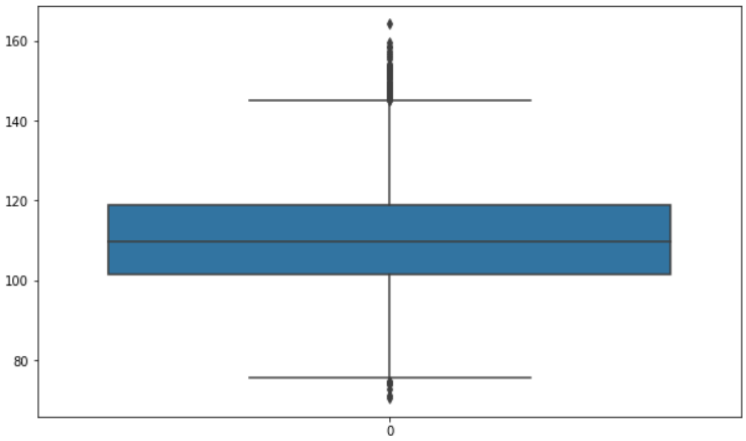

Figure 6.7: Box plot of MC simulation

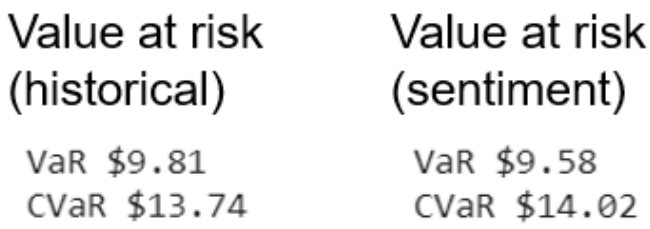

Figure 6.8: Value at risk and Conditional Value at risk of random weights vs. sentiment weights.

I can infer from the results in Figure 6.8 that in terms of Value at risk which is basically the worst possible case within 95% confidence interval, sentiment weights are better, but when considering the absolute worst case which is very unlikely to happen, historical weights can do better.

Chapter 7

Conclusion

Clearly sentiments can be a good way to obtain daily/weekly/monthly adjusted dynamic portfolios. Rather than coming up without any intuition of assigning weights to the portfolio, it is useful to have the sentiment scores calculated as a reflection of recent trends of the stock market. Over a longer period of time, it becomes harder to obtain news data and so it also has its limitations. Also in my case I are limited to only obtaining seven days of ticker data and less than half a month of news data. But given that news can reflect up to 20 days of stock price it is manageable.

Options pricing that are traditionally defined can be looked at a different light after considering the results I obtain after testing out the methodology that I propose in order to determine call option prices and put option. This initial study does not show the result of doing random weight simulation multiple times. Which when performed will show the consistency of the Sentiment Monte Carlo model vs. the Random Monte Carlo simulation which will not change depending on the weight-age of the stocks rather be dependent on getting multiple simulations across multiple random weights which makes it computationally a heavy task even for today's computers.

I run only 4 stocks here. But in real life. A single portfolio can contain over 30 stocks. Running a single instance of sentiment weight will be more feasible rather than

running multiple instances of random weights. Further granular applications of the same need to be performed in future. Right now it is using the average score obtained from a particular day. In future the hourly or 30 minute data intervals can be utilized to create further variations for predictions in a shorter time frame than what is used now.

Appendix A

Source Code for implementation

```
# Import libraries
from urllib.request import urlopen, Request
from bs4 import BeautifulSoup
import os
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
# NLTK VADER for sentiment analysis
from nltk.sentiment.vader import SentimentIntensityAnalyzer

finviz_url = 'https://finviz.com/quote.ashx?t='
yahoo_url = 'https://in.finance.yahoo.com/quote/' #IBN  /?p=  IBN'
reuters_url = 'https://in.reuters.com/companies/' #BJFN.NS  /news'
investing_url = 'https://in.investing.com/search/?q=' #BJFN.NS
↪  &tab=news

news_tables = {}
```

```
tickers = ['AMZN','GOOG','TSLA','AAPL']

for ticker in tickers:
    url = finviz_url + ticker
    #url_yah = yahoo_url + ticker + '/?p=' + ticker
    #print(url)
    req = Request(url,headers={'user-agent': 'Mozilla/5.0'})
    response = urlopen(req)
    # Read the contents of the file into 'html'
    html = BeautifulSoup(response)
    # Find 'news-table' in the Soup and load it into 'news_table'
    news_table = html.find(id='news-table')
    # Add the table to our dictionary
    news_tables[ticker] = news_table
#print(news_table)

# Read one single day of headlines for 'AMZN'
amzn = news_tables['AMZN']
# Get all the table rows tagged in HTML with <tr> into 'amzn_tr'
amzn_tr = amzn.findAll('tr')

for i, table_row in enumerate(amzn_tr):
    # Read the text of the element 'a' into 'link_text'
    a_text = table_row.a.text
    # Read the text of the element 'td' into 'data_text'
    td_text = table_row.td.text
    # Print the contents of 'link_text' and 'data_text'
```

```
        print(a_text)
        print(td_text)
        # Exit after printing 4 rows of data
        if i == 3:
            break

parsed_news = []

# Iterate through the news
for file_name, news_table in news_tables.items():
    # Iterate through all tr tags in 'news_table'
    for x in news_table.findAll('tr'):
        # read the text from each tr tag into text
        # get text from a only
        text = x.a.get_text()
        # splite text in the td tag into a list
        date_scrape = x.td.text.split()
        # if the length of 'date_scrape' is 1, load 'time' as the only
        ↪  element

        if len(date_scrape) == 1:
            time = date_scrape[0]

        # else load 'date' as the 1st element and 'time' as the second
        else:
            date = date_scrape[0]
            time = date_scrape[1]
```

```
        # Extract the ticker from the file name, get the string up to
        ↪  the 1st '_'
        ticker = file_name.split('_')[0]

        # Append ticker, date, time and headline as a list to the
        ↪  'parsed_news' list
        parsed_news.append([ticker, date, time, text])

parsed_news

import nltk
nltk.download('vader_lexicon')

# Instantiate the sentiment intensity analyzer
vader = SentimentIntensityAnalyzer()

# Set column names
columns = ['ticker', 'date', 'time', 'headline']

# Convert the parsed_news list into a DataFrame called
↪  'parsed_and_scored_news'
parsed_and_scored_news = pd.DataFrame(parsed_news, columns=columns)

# Iterate through the headlines and get the polarity scores using vader
scores =
↪  parsed_and_scored_news['headline'].apply(vader.polarity_scores).tolist()
```

```
# Convert the 'scores' list of dicts into a DataFrame
scores_df = pd.DataFrame(scores)

# Join the DataFrames of the news and the list of dicts
parsed_and_scored_news = parsed_and_scored_news.join(scores_df,
↪  rsuffix='_right')

# Convert the date column from string to datetime
parsed_and_scored_news['date'] =
↪  pd.to_datetime(parsed_and_scored_news.date).dt.date

parsed_and_scored_news.head()

parsed_and_scored_news.info()

parsed_and_scored_news.sort_values(by=['date','time'], inplace=True)
parsed_and_scored_news.to_excel("ticker_sentiment.xlsx")

plt.rcParams['figure.figsize'] = [10, 6]

# Group by date and ticker columns from scored_news and calculate the
↪  mean
mean_scores = parsed_and_scored_news.groupby(['ticker','date']).mean()

# Unstack the column ticker
mean_scores = mean_scores.unstack()
```

```
# Get the cross-section of compound in the 'columns' axis
mean_scores = mean_scores.xs('compound', axis="columns").transpose()

# Plot a bar chart with pandas
mean_scores.plot(kind = 'bar')
plt.grid()

# Grouped sentiment scores averaged by date.

import datetime
dailyavgsent=parsed_and_scored_news.groupby(['ticker','date']).mean()
print(dailyavgsent)
#print(list(dailyavgsent.index))

# Filter by a particular desired date
from datetime import date
date_entry = "2021-04-08"
year, month, day = map(int, date_entry.split('-'))
date1 = datetime.date(year, month, day)
print(date1)

extract=dailyavgsent.loc[dailyavgsent.index.get_level_values(level =
↪  'date') == date1]
print(extract)
extracted=extract['compound'].to_numpy()
extracted
```

```
cov_sent=extract['compound']
cov_sent

# Extraction of weights from sentiment scores and standardization using
↪  softmax function (concept in deep learning)

#manual softmax
#def softmax(x):
#    e_x = np.exp(x - np.max(x))
#    return e_x / e_x.sum(axis=0)

# Weights

from scipy.special import softmax
sent_weights=softmax(extracted)
print(sent_weights)

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import datetime as dt
from pandas_datareader import data as pdr

# import data
def get_data(stocks, start, end):
    stockData = pdr.get_data_yahoo(stocks, start, end)
    stockData = stockData['Close']
```

```
    returns = stockData.pct_change()
    meanReturns = returns.mean()
    covMatrix = returns.cov()
    return meanReturns, covMatrix

#stockList = ['CBA', 'BHP', 'TLS', 'NAB', 'WBC', 'STO']
#stocks = [stock + '.AX' for stock in tickers]
endDate = dt.datetime.today()
startDate = endDate - dt.timedelta(days=300)

meanReturns, covMatrix = get_data(tickers, startDate, endDate)

#print(covMatrix)

weights = np.random.random(len(meanReturns))
weights /= np.sum(weights)
print(weights)

# Monte Carlo Method

def MC(weights):
  global initialPortfolio

  mc_sims = 10000 # number of simulations
  T = 30 #timeframe in days

  meanM = np.full(shape=(T, len(weights)), fill_value=meanReturns)
```

```
    meanM = meanM.T

    portfolio_sims = np.full(shape=(T, mc_sims), fill_value=0.0)

    initialPortfolio = 100

    for m in range(0, mc_sims):
        Z = np.random.normal(size=(T, len(weights)))#uncorrelated RV's
        L = np.linalg.cholesky(covMatrix) #Cholesky decomposition to Lower
        ↪ Triangular Matrix
        dailyReturns = meanM + np.inner(L, Z) #Correlated daily returns
        ↪ for individual stocks
        portfolio_sims[:,m] = np.cumprod(np.inner(weights,
        ↪ dailyReturns.T)+1)*initialPortfolio

    return portfolio_sims

portfolio_sims_hist=MC(weights)

plt.plot(portfolio_sims_hist)
plt.ylabel('Portfolio Value ($)')
plt.xlabel('Days')
plt.title('MC simulation of a history based stock portfolio')
plt.show()

portfolio_sims_sent=MC(sent_weights)
```

```
plt.plot(portfolio_sims_sent)
plt.ylabel('Portfolio Value ($)')
plt.xlabel('Days')
plt.title('MC simulation of a sentiment based stock portfolio')
plt.show()

sim_val_hist=portfolio_sims_hist.copy()
sim_val_sent=portfolio_sims_sent.copy()

frame=len(portfolio_sims_hist[0])
sims=len(portfolio_sims_hist)

def call(c):
  for i in range(sims):
    for j in range(frame):
      c[i][j]= c[i][j]-initialPortfolio

      if c[i][j]<0:
        c[i][j]= 0
      else:
        c[i][j]=c[i][j]
  return c

call_hist=call(sim_val_hist)
call_sent=call(sim_val_sent)
#print(call_hist, call_sent)
```

```
sim_val_hist=portfolio_sims_hist.copy()
sim_val_sent=portfolio_sims_sent.copy()

def put(p):
  for i in range(sims):
    for j in range(frame):
      p[i][j]= initialPortfolio-p[i][j]

      if p[i][j]<0:
        p[i][j]= 0
      else:
        p[i][j]=p[i][j]
  return p
put_hist=put(sim_val_hist)
put_sent=put(sim_val_sent)
#print(put_hist, put_sent)

payoff_call_hist=np.mean(call_hist)
payoff_call_sent=np.mean(call_sent)
payoff_put_hist=np.mean(put_hist)
payoff_put_sent=np.mean(put_sent)

print("Call historical payoff:",payoff_call_hist, "\nCall sentiment
↪  payoff:", payoff_call_sent, "\nPut historical payoff:",
↪  payoff_put_hist,
      "\n Put sentiment payoff:", payoff_put_sent)
```

```
def mcVaR(returns, alpha=5):
    """ Input: pandas series of returns
        Output: percentile on return distribution to a given confidence
↪   level alpha
    """
    if isinstance(returns, pd.Series):
        return np.percentile(returns, alpha)
    else:
        raise TypeError("Expected a pandas data series.")

def mcCVaR(returns, alpha=5):
    """ Input: pandas series of returns
        Output: CVaR or Expected Shortfall to a given confidence level
↪   alpha
    """
    if isinstance(returns, pd.Series):
        belowVaR = returns <= mcVaR(returns, alpha=alpha)
        return returns[belowVaR].mean()
    else:
        raise TypeError("Expected a pandas data series.")

portResults = pd.Series(portfolio_sims_hist[-1,:])

VaR_hist = initialPortfolio - mcVaR(portResults, alpha=5)
CVaR_hist = initialPortfolio - mcCVaR(portResults, alpha=5)

print('VaR ${}'.format(round(VaR_hist,2)))
```

```
print('CVaR ${}'.format(round(CVaR_hist,2)))

portResults = pd.Series(portfolio_sims_sent[-1,:])

VaR_sent = initialPortfolio - mcVaR(portResults, alpha=5)
CVaR_sent = initialPortfolio - mcCVaR(portResults, alpha=5)

print('VaR ${}'.format(round(VaR_sent,2)))
print('CVaR ${}'.format(round(CVaR_sent,2)))

import seaborn as sns
sns.boxplot(data=portfolio_sims_sent[-1,:])

plt.rcParams["figure.figsize"] = (15,8)
plt.hist(portfolio_sims_hist[-1,:], bins=50)
plt.grid(True)
plt.xlabel('predicted prices')
plt.ylabel('frequency')

plt.rcParams["figure.figsize"] = (15,8)
plt.hist(portfolio_sims_sent[-1,:], bins=50)
plt.grid(True)
plt.xlabel('predicted prices')
plt.ylabel('frequency')
```

References

[1] Z. Ben Ami and R. Feldman, "Event-based trading: Building superior trading strategies with state-of-the-art information extraction tools," *Available at SSRN 2907600*, 2017.

[2] L. A. Smales, "The importance of fear: investor sentiment and stock market returns," *Applied Economics*, vol. 49, no. 34, pp. 3395–3421, 2017.

[3] P. Hafez and J. Xie, "Web news analytics enhance stock portfolio returns," *Available at SSRN 2423362*, 2014.

[4] W. S. Leung, G. Wong, and W. K. Wong, "Social-media sentiment, portfolio complexity, and stock returns," *Portfolio Complexity, and Stock Returns (November 24, 2019)*, 2019.

[5] D. de França Costa and N. F. F. da Silva, "Inf-ufg at fiqa 2018 task 1: predicting sentiments and aspects on financial tweets and news headlines," in *Companion Proceedings of the The Web Conference 2018*, 2018, pp. 1967–1971.

[6] C.-J. Wang, M.-F. Tsai, T. Liu, and C.-T. Chang, "Financial sentiment analysis for risk prediction," in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 2013, pp. 802–808.

[7] M. S. Akhtar, A. Kumar, D. Ghosal, A. Ekbal, and P. Bhattacharyya, "A multilayer perceptron based ensemble technique for fine-grained financial sentiment analysis," in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, pp. 540–546.

[8] N. C. Dang, M. N. Moreno-García, and F. De la Prieta, "Sentiment analysis based on deep learning: A comparative study," *Electronics*, vol. 9, no. 3, p. 483, 2020.

[9] A. Martin-Utrera, "Shrinking against sentiment: Exploiting behavioral biases in portfolio optimization," *Available at SSRN 3551224*, 2020.

[10] H. K. Sul, A. R. Dennis, and L. Yuan, "Trading on twitter: Using social media sentiment to predict stock returns," *Decision Sciences*, vol. 48, no. 3, pp. 454–488, 2017.

[11] A. Bandopadhyaya, A. L. Jones *et al.*, "Measures of investor sentiment: A comparative analysis put-call ratio vs. volatility index," *Journal of Business & Economics Research (JBER)*, vol. 6, no. 8, 2008.

[12] M. Baker and J. Wurgler, "Investor sentiment in the stock market," *Journal of economic perspectives*, vol. 21, no. 2, pp. 129–152, 2007.

[13] C. Yang, B. Gao, and J. Yang, "Option pricing model with sentiment," *Review of Derivatives Research*, vol. 19, no. 2, pp. 147–164, 2016.

[14] R. Hao, "Option pricing model with investor sentiment," 2017.

[15] B. Han, "Investor sentiment and option prices," *The Review of Financial Studies*, vol. 21, no. 1, pp. 387–414, 2008.