

Final Report



Building a Repository for Teaching Algorithmic Trading

Wu Xue, Snow

(3035372787)

COMP 4801 Final year project

Supervisor: Dr. Rui Bang Luo

April 18, 2021

Abstract

Trading is complicated and requires time, attention, and resources to study the market to make a rational decision. With the improvement of technology, algorithms evolved tremendously over the years allowing for less supervision and freedom. However, learning algorithmic trading for non-institutional investors has failed to keep up with the progress. This paper builds up an open-source and data repository that 1) contains pipelines for collecting a firm's stock price, company financial statements, average monthly property price, financial news, and social media data. 2) implemented trading signal using machine learning combining microeconomic, macroeconomic and sentiment analysis in python and evaluated with the backtester feature in the repository. 3) Set up a paper trading account by leveraging the trained model to carry out daily trading signal and directly sent to the interactive broker API 4) Documented all code implementations and created tutorials that enable investors to learn algo trading regardless of their background knowledge in finance and coding. Experiments have been conducted from 2017-01-03 to 2021-03-03 of Hong Kong Stock Exchange data exploring different machine learning and baseline model. The results show that the baseline model has the best performance with a mean cumulative portfolio return of 5.5% in the from June 2020 to March 2021.

The repository and the documentation website are respectively available at <https://github.com/awoo424/algotrading> and <https://algo-trading.readthedocs.io/>.

Acknowledgements

I want to express my deep gratitude towards our supervisor, Dr. R.P. Luo, for his support and advice. Without those, our project would not have been possible.

I would also like to express my thanks to my teammates for their active devotion to the projects. We have carried out a regular meeting every week and actively discussed the different difficulties we faced and gave recommendations to each other.

Table of Content

Abstract.....	i
Acknowledgements.....	ii
Table of Content	iii
List of Figures	v
List of Tables	vi
List of Algorithms	vi
Abbreviations and Acronyms	vii
1. Introduction	1
2. Contributions	2
2.1 Education Contributions	2
2.2 Research Contributions.....	3
2.3 Scope of the work	3
2.4 Report outline	4
3. Methodology.....	4
3.1 Data collection	4
3.1.1 Data collection of Microeconomic data.....	4
3.1.2 Data collection of Macroeconomic data.....	8
3.1.3 Data collection of Sentimental data	9
3.1.3.1 Data collection and Updating of financial News Headline.....	9
3.1.3.2 Data collection and Updating of historical tweets.....	11
3.2 Data Analysis.....	12
3.2.1 Data Analysis on Microeconomic data.....	12
3.2.2 Data Analysis on Macroeconomic data.....	19
3.3 Machine learning predictions	22
3.3.1 Bankruptcy prediction.....	22
3.3.2 Property Price Prediction	24
3.3.3 Sentimental analysis using Vader library	25
3.3.4 Sentimental analysis using Text blob library.....	27
3.3.5 Sentimental analysis using Loughran-McDonald Financial Dictionary 2018	27
4. Experiments	28
4.1 Datasets	28
4.1.1 Datasets from Sentiment factor	29
4.1.2 Datasets from Technical factor	30
4.1.3 Datasets from macroeconomic factor	31
4.2 Baseline Setup.....	32

4.3	LSTM setup.....	33
4.3.1	LSTM single feature model	33
4.3.2	LSTM multiple feature model	34
5	Results.....	34
5.1	Baseline Model.....	34
5.2	LSTM single feature price results.....	35
5.3	LSTM multiple feature absolute price results.....	36
5.4	Conclusion on the model exploration.....	38
5.5	Documentation of the website	38
6	Limitations.....	39
6.1	Limitations in web scraping technologies.....	39
6.2	limitation in API rate limit:.....	40
6.3	Limitation in the amount of data collected.	40
7	Conclusion and Future works.....	40
	References	1

List of Figures

Figure 1 : Overall workflow of the whole project	3
Figure 2: Top 4 rows of the entire list of traded tickers in HKEX.	4
Figure 3: Top 4 rows in hkex_0001.csv that contains the output of the 1-day tick of ticker "0001" in HKEX.	5
Figure 4: Example of table storing fundamentals data for a stock on Yahoo! Finance page. ...	7
Figure 5: HTML document containing the real-time price in Yahoo! Finance page.	7
Figure 6: The data structure of transaction record from Centaline Property.	8
Figure 7: The data structure of transaction record from Midland Realty- Part 1.	9
Figure 8: The data structure of transaction record from Midland Realty- Part 2.	9
Figure 9: Line plot illustrates the relationship of time span with portfolio return on MACD crossover strategy.....	15
Figure 10: Line plot illustrates the relationship of time span with the number of trade signals on MACD crossover strategy.....	15
Figure 11: Summary statistics of applying MACD on all tickers in HKEX from the given time span.	16
Figure 12: Heat map illustrates the number of trading signals under different strategies using 0005.HK from 2017 to 2019.....	16
Figure 13: Heat map illustrates the performance of Sharpe ratio under different strategies using 0005.HK from 2017 to 2019.	17
Figure 14: Heat map that illustrates the performance of Sharpe ratio for different strategies from the given time span using 0005.HK.....	18
Figure 15: Regression output of Average Price saleable area with the adjusted unemployment.....	20
Figure 16: Correlation heatmap showing the correlation coefficient in the property transaction data.....	22
Figure 17: The graph of predicted and actual house price using XGBoost model.	25
Figure 18: Figure shows the Vader label of tweets samples from NTLK library.....	26
Figure 19: High level workflow of LSTM single feature model.....	33
Figure 20: High level workflow of LSTM multi-feature workflow	34
Figure 21: Bar graph shows the portfolio return from 2020-06-10 to 2021-03-03.	35
Figure 22: Bar graph shows the portfolio return from 2020-06-10 to 2021-03-03 under LSTM single-feature model.....	36

Figure 23:Bar graph shows the portfolio return from 2020-06-10 to 2021-03-03 under LSTM multi-feature model absolute price result.	37
---	----

List of Tables

Table 1:List of technical indicators covered in the repository.....	13
Table 2: Table shows the correlation coefficient between the economic indicators and the monthly average house price per saleable area in Hong Kong.....	21
Table 3: List of commonly used financial ratios in fundamental analysis.	23
Table 4: Accuracy score of different machine learning models in predicting 1-year and 3-year bankruptcy.....	24
Table 5: Table with the RMSLE value of different models.....	25
Table 6: Table shows the list of selected tickers - part 1	29
Table 7: Table shows the list of selected tickers - part 2	29
Table 8: Table shows the Vader and TextBlob sentiment label for daily news of selected tickers from January 2017 to March 2021	30
Table 9: Table shows the fundamental data for daily information of selected tickers from January 2017 to March 2021.....	30
Table 10:Table shows the mean value of MACD, returns and signal in each industry.....	31
Table 11: Table shows different indicators under macroeconomic aspect.....	31
Table 12: Table shows the summary statistics in the macroeconomic data	32
Table 13:Table shows the summary statistics under the baseline model from 2020-06-10 to 2021-03-03.....	35
Table 14: Table shows the summary statistics under the LSTM single-feature model from 2020-06-10 to 2021-03-03.	36
Table 15: Table shows the summary statistics under the LSTM multi-feature model from 2020-06-10 to 2021-03-03.	37
Table 16: Table taken from macrotrends.net showing the list of Historical Annual Data from 2017 to 2021	38

List of Algorithms

Algorithm 1: Pseudo code for downloading the stock tick data	5
--	---

Algorithm 2: Pseudo code to update the stock tick data.....	6
Algorithm 3: Pseudo code to scrape fundamental data from Yahoo! Finance page.....	7
Algorithm 4: Pseudo code to retrieve the real-time stock price.....	8
Algorithm 5: Code snippet for updating news from finviz.com.....	10
Algorithm 6: Code snippet for updating news from aastock.com.....	11
Algorithm 7: Code snippets for retrieving tweets from the relevant hashtag.....	12
Algorithm 8: Pseudo code to backtest a technical analysis strategy.....	14
Algorithm 9: Pseudo code to carry out stock screening.....	18
Algorithm 10: Code snippet for carrying out the univariate analysis.....	19
Algorithm 11: Pseudo code to retrieve the Vader score.....	26
Algorithm 12: Pseudo code to retrieve the TextBlob score.....	27
Algorithm 13: Pseudo code for the threshold function.....	27
Algorithm 14: Pseudo code to retrieve the Loughran label.....	28
Algorithm 15: Pseudo code for the threshold function in Loughran function.....	28
Algorithm 16: Code for updating signals after the macroeconomic filter.....	32

Abbreviations and Acronyms

AI	Artificial Intelligence
NLP	Natural Language Processing
LSTM	Long short-term memory
NLTK	Natural Language Toolkit
URL	Uniform Resource Locator
API	Application programming interface
Algo	Algorithmic
HK	Hong Kong
US	United States
HKEX	Hong Kong Stock Exchange
NYSE	New York Stock Exchange
Mkt	Market
HSI	Hang Seng Index

1. Introduction

Algorithmic trading is commonly defined as using computer programs to make auto trading decisions, submit orders, and oversee these orders after submission (Hendershott et al.,2009). Investors must convert their trading strategies into computer code or mathematical equations for the computers to proceed on executing the trade orders based on the prescribed instructions. By now, the market share of algorithmic has gauged up to 90% of the total trading in the market. (Melin,2017). Under such a paradigm, it is not surprising to see that big banks, hedge funds, and institutional investors have all shifted to using algo trading to execute a trading opportunity in the market.

Furthermore, algo trading becomes a prerequisite for surviving in tomorrow's financial markets due to the automation trend in the future of trading and dealing. The market of algo trading is expected to grow from \$11.1 billion in 2019 to 18.8 billion by 2024 with a compound annual growth rate (CAGR) of 11.1%. (Aggarwal,2019)

Many previous research works focused on one aspect of market information, such as using sentiment analysis from financial news to predict stock price (Schumaker and Chen, 2009). While Li, Audeliano, Bastos and Guilherme (2008) exploit the deep learning models by processing the firm's historical data and technical indicators to predict stock prices. On top of that, Tse (2001) studies the impact of real estate prices on the typical stock prices in Hong Kong as Real-estate account for over 30% of Hong Kong's stock market capitalization. However, to the best of our knowledge, only a few research works deal with multiple aspects of market information on macroeconomic, microeconomic and sentiment data at the same time to predict the stock prices.

Another issue of the current situation in the market is that though many institutional investors are taking an active approach to carry out algo trading, individual investors face a different fate. Information needed for algo trading has been available but uneasy for individual investors due to lack of knowledge in technology and trading experience. Also, a large amount of data flow across various sources and the need to adjust strategies have challenged their cognitive limits and place them at a disadvantage. By studying from different learning resources available

(including blogs posts, and websites) on the internet, we concluded that they suffer from the following problems in general:

- 1) Information is scattered and usually only covers a few concepts about algorithmic trading.
- 2) Single-sided focus on either finance or programming concept
- 3) Majority of online resources merely focus on the US market and do not fit the local context.
- 4) No practical tutorials on converting strategies to real-life trading

To address those issues, we propose an approach in this paper that 1) produce an all-in-one, comprehensive educational resource that puts together the interdisciplinary knowledge necessary for learning algo trading (i.e. data science, coding, and fundamental finance concepts). 2) Align with the financial setting of Hong Kong and providing data on the US market at the same time. 3) Set up paper trading account to automate the stock ordering system.

We design and conduct experiments integrating data with five years of Hong Kong Stock Exchange from our built data repository. We have tested different models, including the baseline (different market indicators act as filters to filter out daily signals) and LSTM multi-feature model (taking market indicators as machine learning input features). We find that the baseline model has the highest performance with a mean portfolio return of around 5.5% from the experimental results.

2. Contributions

This project aims to achieve several educational and research objectives to server as algorithmic trading teaching materials and stock movement predictions based on multi-source features.

2.1 Education Contributions

The primary goal of this educational objective is to build a data repository that brings data science skills, financial concepts, and algorithmic design techniques that are vital to learning Algo Trading. We hope the information is clear and understandable to the user without going back and forth on different websites to extract useful information. The repository content also serves as a library to appeal to a broad audience who have little or no knowledge in finance but are interested in having a clear understanding of how to translate the financial concept into computer code.

2.2 Research Contributions

The research objectives aim to integrate different indicators collected from the microeconomic, macroeconomic, and market sentiment to form more comprehensive stock predictions to investors in the Hong Kong and US markets. Also, we have constructed a consolidated database consisting of the historical stock tick, property price, Twitter feeds, financial news related to each listed company. This data repository serves as a resource for future research purposes at the University of Hong Kong to test different trading strategies.

2.3 Scope of the work

The project is divided into two main parts, as shown in Figure 1. The first parts focus on building an open data repository that is essential for forming a trading strategy. We have examined data from three perspectives- microeconomic, macroeconomic, and sentimental. Concerning each market aspect, historical data were collected, and trading indicator was created to analyze the financial market,

In the second part, we have implemented different LSTM models and also baseline model that leverage data from the above three perspectives to predict stock price movement. Additionally, simulated virtual investment was experimented with by opening a Paper Trading account in the interactive broker (IB). A final product contains a repository and relevant code snippets, which consists of traditional data and alternative data to build, test, and evaluate an algo trading strategy.

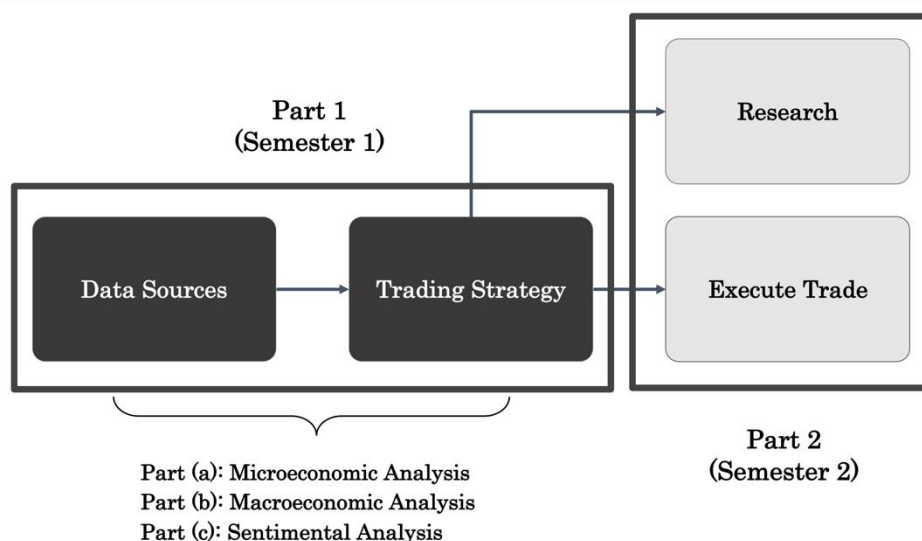


Figure 1 : Overall workflow of the whole project

2.4 Report outline

The remaining of this part is organized as follows. Chapter 3 discusses different methodologies for retrieving data from the macroeconomic, macroeconomics, and sentiment market. This Chapter also covers data analysis and machine learning predictions to get trading signals from three different aspects. Chapter 4 then describes the experimental setup and related datasets for the baseline and the LSTM model. Chapter 5 discusses the results from the baseline and the LSTM model. A summary of different significant limitations was provided in Chapter 6. Chapter 7 sums up the project, and a conclusion was made.

3. Methodology

Different methodologies were implemented to collect data from the three market perspectives. In the following, we will elaborate on the various methodologies carried out for each subpart from the data collection and data analysis perspective. After that, machine learning predictions were carried out on predicting a firm's bankruptcy probability, monthly property price and market sentiment.

3.1 Data collection

3.1.1 Data collection of Microeconomic data

Stock tick data. We have downloaded a list of listed tickers from the stock exchange's official websites. The list is load into the Jupyter Notebook as an array and traverses down it to download all the data for each ticker by calling the Yahoo! Finance API. Figure 2 shows the snapshot of the ticker list in HKEX, and the outline of the algorithm for downloading the stock tick data is shown in Algorithm 1.

	symbol	name
0	0001	Cheung Kong (Holdings) Ltd
1	0002	CLP Holdings Ltd
2	0003	Hong Kong and China Gas Co. Ltd
3	0004	Wharf (Holdings) Ltd
4	0005	HSBC Holdings plc

Figure 2: Top 4 rows of the entire list of traded tickers in HKEX.

Algorithm 1: Download stock tick data

Data: List of stock ticker symbols**Result:** csv files with columns (Date, Open, High, Low, Close, Volume)

```

1 Import pandas and yfinance;
2 Load list of symbols into an array  $S[1..n]$ ;
3 for symbol in  $S[1..n]$  do
4   |  $data \leftarrow$  Call the Yahoo! finance API;
5   |  $df \leftarrow$  pandas.DataFrame(data);
6   | Export df to a csv file;
7 end

```

Algorithm 1: Pseudo code for downloading the stock tick data

Daily data with variables with floating-point numbers: Date, Open, High, Low and Close columns were created. Figure 3 shows an example output CSV file of the stock tick data for a ticker.

	Date	Open	High	Low	Close	Volume
0	2000-01-04	33.037451	33.367817	32.376717	32.376717	3194413
1	2000-01-05	30.890019	31.385591	29.981523	30.146683	6058531
2	2000-01-06	30.394473	30.559633	28.081818	28.659994	10440480
3	2000-01-07	29.072963	29.403330	28.577392	29.238123	6049796
4	2000-01-10	30.229264	30.724838	29.485929	29.485929	5195405

Figure 3: Top 4 rows in `hkex_0001.csv` that contains the output of the 1-day tick of ticker "0001" in HKEX.

The 1-day price data for each ticker were downloaded since the company's IPO, while the 1-minute data were downloaded from 1 June 2020. Algorithm 2 shows the pseudocode to automate downloading data from the last updated date and the latest trading day.

Algorithm 2: Update stock tick data (in-place)

Data: Directory containing stock tick files
Result: Stock tick files with records til latest trading day

```

1 Import modules;
2 dir_name ← name of directory storing the stock tick files;
3 pathlist ← Path(dir_name).rglob('*.csv');
4 for path in pathlist do
5     df ← Read file as csv;
6     last_date ← Get last date in df;
7     today ← Get today's date;
8     dates ← list of dates between last_date and today;
9     if last_date != today then
10        data ← Call Yahoo! Finance API with start=dates[0], end=dates[-1];
11        if last_date == last date in data then
12            # Today is not a trading day
13            print(Data already up-to-date, no changes made);
14        else
15            Append data to df;
16        end
17    else
18        print(Data already up-to-date, no changes made);
19    end
20 end
21 print success message;
```

Algorithm 2: Pseudo code to update the stock tick data


We have run the following command to update the stock tick data for a particular stock exchange:

```
make update-<place/stock exchange abbreviation>-<day or min>
```

As an illustration, the 1-minute price data listed in HKEX were downloaded using: make update-hk-min. A success message would be displayed.

Firm's Fundamental data. Due to the absence of an existing API, we have collected fundamental data through web-scraping using the lxml library to parse the HTML document. Figure 4 shows the table existing on the Yahoo! Finance webpage, and Algorithm 3 shows the outline of the algorithm for scrapping fundamentals data.

Income Statement All numbers in thousands

 Get access to 40+ years of historical data with Yahoo Finance Premium. [Learn more](#)

Breakdown	TTM	12/30/2019	12/30/2018	12/30/2017
> Total Revenue	276,052,000	299,021,000	277,129,000	248,515,000
Cost of Revenue	122,513,000	131,993,000	129,811,000	101,328,000
Gross Profit	153,539,000	167,028,000	147,318,000	147,187,000
> Operating Expense	112,374,000	115,372,000	102,345,000	111,792,000
Operating Income	41,165,000	51,656,000	44,973,000	35,395,000
> Net Non Operating Interest Inc...	-12,772,000	-14,305,000	-9,797,000	-8,274,000

Figure 4: Example of table storing fundamentals data for a stock on Yahoo! Finance page.

Algorithm 3: Scrape fundamentals data

Data: List of stock ticker symbols

Result: csv files with fundamentals data

```

1 Import modules;
2 Load list of symbols into an array  $S[1..n]$ ;
3 for symbol in  $S[1..n]$  do
4   page ← Fetch the Yahoo! Finance page for symbol;
5   tree ← Parse the page with lxml;
6   table_rows ← Fetch all div elements which have class  $D(tbr)$ ;
7   assert len(table_rows) > 0;
8   df ← parse table_rows;
9   df ← clean the data in df;
10  Export df to a csv file;
11 end

```

Algorithm 3: Pseudo code to scrape fundamental data from Yahoo! Finance page

Real-time prices. Though it is not collected as part of the repository database, we acknowledge that it is still useful in more specialized fields such as intra-day trading and high-frequency trading. Hence, we provide a code example below to demonstrate how to scrape real-time prices for a specific symbol using the BeautifulSoup library.

```

<span class="Trsdu (0.3s) Fw(b) Fz(36px) Mb(-4px) D(ib)"
data-reactid="32">
  56.000
</span>

```

Figure 5: HTML document containing the real-time price in Yahoo! Finance page.

To scrape the real-time price of 56 as shown in Algorithm 4, we have used the below code to crawl the data:

Algorithm 4: Pseudo code to retrieve the real-time stock price

```

session = requests_html.HTMLSession ()
r = session.get(url) # url = the Yahoo! Finance page with the price
soup = BeautifulSoup (r. content, 'lxml')
try:
    price = soup. select ('table td') [5].text. split(' ')[0]
    price = float(price)
except IndexError as e:
    price = None
  
```

Algorithm 4: Pseudo code to retrieve the real-time stock price

3.1.2 Data collection of Macroeconomic data

Hong Kong residential market transaction records. We have built web scrapers to collect Hong Kong residential market transaction records from two real estate website (Centaline Property and Midland Realty) using web scraping APIs and open-source libraries.

For Centaline Property, the entire HTML page was retrieved after sending the GET request with the district id and registration period parameters. Figure 6 shows the data structure of transaction records extracted from Centaline Property. The data contains ten basic features describing the apartment from 02/01/2017 to 23/12/2020.

address	buildingAge	regDate	price	saleableArea	grossArea	upSaleableArea	upGrossArea	lasthold	gain
FLAT 7 26/F BLOCK A SMITHFIELD TERRACE	34	2020-12-23	5300000.0	262	357	20229	14846	3394	89
FLAT H 41/F THE FOREST HILLS	12	2020-12-23	7380000.0	439	627	16811	11770	2793	50
FLAT 5 (NO. 26) 1/F MAN YUE MANSION (BUILDING)	56	2020-12-23	3980000.0	480	-	8292	-	-	NaN
FLAT 12 30/F CHUN HONG HOUSE (BLOCK E) TIN MA ...	34	2020-12-23	4800000.0	-	461	-	10412	-	NaN
NO. 3 4/F GOLDEN PHOENIX BUILDING	55	2020-12-23	3350000.0	381	-	8793	-	7718	329

Figure 6: The data structure of transaction record from Centaline Property.

For Midland Realty, an array of transaction records was returned after sending the GET request with district id and registration period as the parameter. Figure 7 shows the data structure of transaction records extracted from Midland Realty. The address section was further divided into sub-features and additional features (floor level and the number of bedrooms) compared with the Centaline property data.

address	buildingAge	regDate	price	saleableArea	grossArea	upSaleableArea	upGrossArea	lasthold	gain
FLAT 7 26/F BLOCK A SMITHFIELD TERRACE	34	2020-12-23	5300000.0	262	357	20229	14846	3394	89
FLAT H 41/F THE FOREST HILLS	12	2020-12-23	7380000.0	439	627	16811	11770	2793	50
FLAT 5 (NO. 26) 1/F MAN YUE MANSION (BUILDING)	56	2020-12-23	3980000.0	480	-	8292	-	-	NaN
FLAT 12 30/F CHUN HONG HOUSE (BLOCK E) TIN MA ...	34	2020-12-23	4800000.0	-	461	-	10412	-	NaN
NO. 3 4/F GOLDEN PHOENIX BUILDING	55	2020-12-23	3350000.0	381	-	8793	-	7718	329

Figure 7: The data structure of transaction record from Midland Realty- Part 1.

flat	grossArea	saleableArea	price	regDate	lastRegDate	lastPrice	gain	lat	lon
6	597.0	498.0	8250000.0	2021-01-07	2003-01-17	1700000.0	385.29	22.278636	114.239373
D	395.0	327.0	5400000.0	2021-01-07	NaT	NaN	0.00	22.386003	114.204891
B	1251.0	952.0	21000000.0	2021-01-07	2008-01-18	9680000.0	116.94	22.304435	114.184404
21	629.0	485.0	3950000.0	2021-01-07	NaT	NaN	0.00	22.361853	114.103990
E	883.0	740.0	7830000.0	2021-01-07	1997-08-06	4953900.0	58.06	22.440296	114.034537

Figure 8: The data structure of transaction record from Midland Realty- Part 2.

3.1.3 Data collection of Sentimental data

We have collected the market sentiment data from different platforms such as news articles and Twitter. As the popularity of platforms varies across countries, we had applied different text mining technologies to extract data that has the highest relevance to the local market.

3.1.3.1 Data collection and Updating of financial News Headline

We have implemented various code designs to scrape data from the web using the Beautiful Soup library for accounting for the difference in front-end architecture in different websites.

Regarding the US market, we have collected relevant financial news for each listed company in NASDAQ and NYSE from finviz.com. It is a browser-based stock market research platform that allows visitors to see the latest financial news collected from different primary newsagents such as Yahoo! Finance, Accesswire, and Newsfile.

The code snippet shown in Algorithm 5 demonstrates the program designed for updating news from finviz.com. By passing in the ticker name on a required day, a GET request was sent to the URL in the ticker's page that features all the relevant news. Finally, we extracted news headlines under the div tag with the id name of the news-table and stored them in the ticker's corresponding CSV file. All the relevant code could be found in [../sentiment_analysis/news-finviz_collection.ipynb](#).

Peseudo description of Algorithm 5

Data: ticker name and the require day

Result: relevant csv file containing the finviz news

Extract the HTML code using Beautiful soup library

Loop through the <tr> tag containing the relevant finviz news:

1. Retrieve the text string using `.get_text()` function
 2. Retrieve the corresponding date string using `td.text.split()` funtion
 3. Append the news to the relevant CSV file it is within the date range
-

```
# prefix url to access finviz.com
finviz_url = 'https://finviz.com/quote.ashx?t='
def get_finviz(ticker,day):
    try:
        url = finviz_url + ticker
        req = Request(url=url,headers={'user-agent': 'my-app/0.0.1'})
        resp = urlopen(req)
        html = BeautifulSoup(resp, features="lxml")
        # access the news stored under the <div id='news-tabel'>
        news_table = html.find(id='news-table')
        news_tables[ticker] = news_table
        df = news_tables[ticker]
        path=os.path.join(dir_name,'data-news/data-finviz/'+data+'-finviz.csv')
        # append the news into the corresponding ticker's csv file
        with open(path,'a') as f:
            print(ticker)
            writer = csv.writer(f)
            for info in df.findAll('tr'):
                # get the text of the financial headlines
                text=info.a.get_text()
                date_scrape= info.td.text.split()
                if(len(date_scrape)==1):
                    time=date_scrape[0]
                else:
                    date= date_scrape[0]
                    time=date_scrape[1]
                    news_time_str= date+" "+time;
                # convert the date into datetime object of 'YYYY-mm-dd' format
                date_time_obj = datetime.datetime.strptime(news_time_str, '%b-%d-%y %I:%M%p').strftime('%Y-%m-%d')
                if(datetime.datetime.now()-date_time_obj).days<=day:
                    writer.writerow([date_time_obj,text])
```

Algorithm 5: Code snippet for updating news from finviz.com

Regarding the Hong Kong market, we have obtained financial news headlines relevant to each listed company from aastock.com. The website has been one of the highest-ranking financial information platforms in Hong Kong for more than a decade. It offers real-time information relevant to Hong Kong shares, helpful in analyzing sentiment and trends in the local market.

The code snippet shown in Algorithm 6 demonstrates the code for updating the news from aastock.com. By passing in the ticker name on a required day, a GET request will be sent to the URL in the ticker's page that features all the relevant news. As opposed to data from

finviz.com, note that the news headlines and dates were not well organized in a table form. As headline text and dates were stored in chronological order in separate div elements, text and dates were collected separately and then joined as a table (by matching the Date Time) at the last stage. All the relevant code could be found in [./integrated-strategy/models/collect_news_aastock.py](#).

Peseudo description of Algorithm 6

Data: the path of the specific ticker's CSV file

Retrieve all the dates and text into array name of dates and news

Result: relevant csv file containing the aastock news

Loop through the Date array:

4. Remove the 'Release_Time' prefix (Date text contain dates string with 'Release_Time' as the prefix)
 5. Extract the financial text with the corresponding index
 6. Extract the text if it is within the time range
 7. Append the Date and text into the csv file
 8. Increment the index to the next financial news
-

```
with open(path,action) as f:
    writer = csv.writer(f)
    for i in dates:
        if "/" in str(i.get_text()):
            date = str(i.get_text())
            if "Release Time" in date:
                date = date[13:23]
            elif (date[0] == " "):
                date = str(date[1:11])
            else:
                date = str(date[0:10])
            # print(date)
            text = news[idx].get_text()
            # get date
            date_time_obj = datetime.datetime.strptime(date, '%Y/%m/%d')
            date_time = date_time_obj.strftime('%Y-%m-%d')
            date_now = datetime.datetime.now().strftime('%Y-%m-%d')

            if (datetime.datetime.now()-date_time_obj).days <= days:
                print(text)
                # write to file
                writer.writerow([date_now,text,ticker,newstype])
            idx += 1
```

Algorithm 6: Code snippet for updating news from aastock.com

3.1.3.2 Data collection and Updating of historical tweets

Historical tweets. We have collected historical tweets about each company listed in the US (NYSE and NASDAQ) through the Tweepy API. The company's ticker symbol was passed as the API input to retrieve the tweets with hashtags mentioning the stock. Corresponding tweets and dates were collected and appended to the company's CSV file.

The code in Algorithm 7 indicates the function for updating tweets from Twitter. By passing in the ticker name on a required day, its relevant tweets were presented in the extended mode (i.e. having the full text of the tweets). The data was then updated to the data repository after the text has been encoded in the utf-8 mode. All the relevant code could be found in [../sentiment_analysis/tweets_collection.ipynb](#).

Peseudo description of Algorithm 7

Data: Stock tick data for a specific ticker

Result: relevant csv file containing the tweets

9. Hashtag cursor \leftarrow Connect to the API using the cursor
 10. file f \leftarrow Open the relevant csv file for a specified ticker symbol
 11. loop through the Hashtag cursor :
 12. extract the tweets from the full text using `.full_text` function if it is within the time range
 13. Encode the tweets into 'utf-8' mode
 14. Append the tweets and Date into the csv file
-

```
# get data from the cashtag
def get_tagtweets(name, day):
    allhashtag=[]

    hashtags=tweepy.Cursor(api.search,q=name,lang='en',tweet_mode='extended').items(300)
    outtags=[]
    try:
        path=os.path.join(dir_name, 'data-tweets/')
        with open(path+'data-'+name+'-tweets.csv', 'a') as f:
            writer = csv.writer(f)
            for status in hashtags:
                if(datetime.datetime.now()-status.created_at).days<=day:
                    tweet_text = status.full_text.encode("utf-8")
                    dates=str(status.created_at)[:10]

                    print(dates)
                    print(tweet_text)
                    writer.writerow([dates,tweet_text])
            sleep(2000)
    except:
        print("error")
        pass
```

Algorithm 7: Code snippets for retrieving tweets from the relevant hashtag

3.2 Data Analysis

3.2.1 Data Analysis on Microeconomic data

We aim to cover four different trading scenarios under the technical analysis to illustrate how users can use our backtester, indicators, and historical data in the repository to perform research and analyse trading strategies.

3.2.1.1 Technical Analysis

We have implemented 16 technical indicators from four different categories: trend, momentum, volume, and volatility in python. Every indicator has its python class and comes along with an example code file with a prefix of `main_`. The example file demonstrates how to call the class for each indicator. A backtester has been implemented in python to evaluate the strategy's performance together with the historical data. Table 1 shows the complete list of indicators implemented in the repository. Algorithm 8 shows the major steps of running a technical analysis

Category	Indicator(s)
Trend	<ul style="list-style-type: none"> • Moving Average Crossovers • Moving Average Convergence Divergence (MACD) • Parabolic Stop and Reverse (Parabolic SAR)
Momentum	<ul style="list-style-type: none"> • Commodity Channel Index (CCI) • Relative Strength Index (RSI) • Rate of Change (ROC) • Stochastic Oscillator (STC) • True Strength Index (TSI) • Money Flow Index (MFI) • Williams %R
Volume	<ul style="list-style-type: none"> • Chaikin Oscillator • On-Balance Volume (BOV) • Volume Rate of Change
Volatility	<ul style="list-style-type: none"> • Bollinger Bands • Average True Range (ATR) • Standard Deviation

Table 1: List of technical indicated covered in the repository.

Algorithm 8: Backtest a technical analysis strategy

Data: Stock tick data for a specific ticker

Result: png files of figures generated and backtesting results printed on terminal

15. Import required libraries

-
16. `df` ← Load stock tick data for a specified ticker symbol
 17. Select time range to backtest
 18. `ticker` ← Set ticker symbol to test
 19. `strategy` ← Create an instance of the strategy class by passing `df` as argument
 20. `signals`, `figure` ← Call `gen_signals` and `plot_signals` methods in strategy class
 21. Call the Backtester function by passing `ticker`, `signals` and `df` as arguments
 22. Call evaluation metric functions as needed (e.g Sharpe ratio, maximum drawdown)
-

Algorithm 8: Pseudo code to backtest a technical analysis strategy

All the figure generated by the backtester in the example main files could be found in the `technical-analysis_python/figure` directory. All absolute values, including the final portfolio value and the number of trade signal, will be shown as output in the terminal by running through the above code.

The implementation and interpretation of the trading strategies are essential in real trading situations. We hope to provide four investment scenarios to help investments gain investment insights from different angles from different technical analysis.

Scenario 1: Investigate the optimal period (long-term or short terms) of the investment, given a ticker symbol and a strategy.

- Dependent variables: Portfolio return, Number of trades
- Independent variables: Ticker=0005.HK, Strategy=MACD crossover, Time span=2017-01-01 to 2019-01-01

We have chosen Ticker 0005.HK to be our testing firm. Figure 9 shows that the portfolio's return graph shows that a crest or trough was formed roughly every four years. While Figure 10 shows a negative correlation between the number of trading signals and the duration of the period.

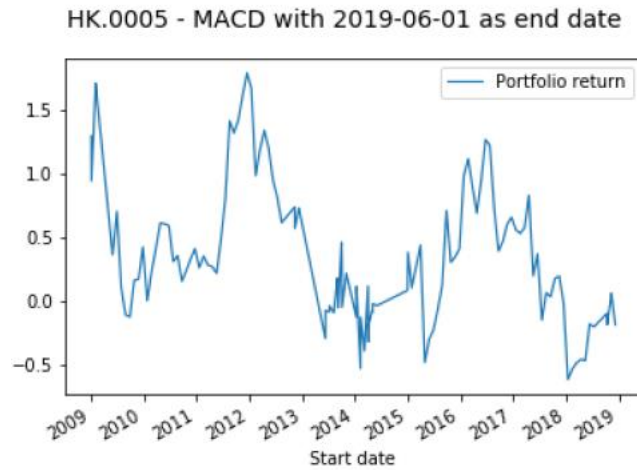


Figure 9: Line plot illustrates the relationship of time span with portfolio return on MACD crossover strategy.

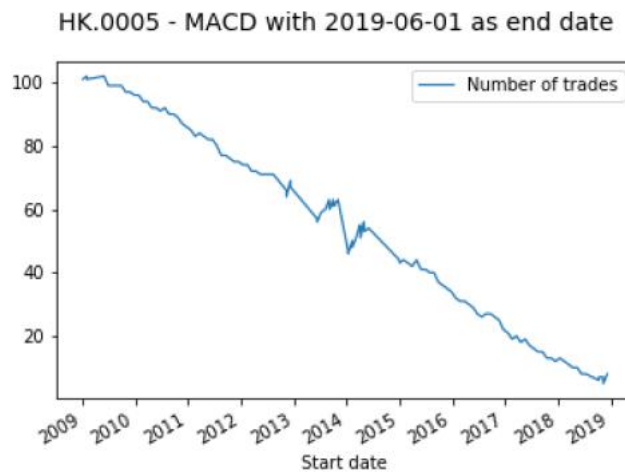


Figure 10: Line plot illustrates the relationship of time span with the number of trade signals on MACD crossover strategy.

Scenario 2: Investigate the general performance of the strategy by back testing on different tickers given a fixed period and a strategy.

- Dependent variables: Portfolio return, Number of trades
- Independent variables: Ticker=0005.HK, Strategy=MACD crossover, Time span=2017-01-01 to 2019-01-01

	Portfolio return	Number of trades
count	1369.000000	1369.000000
mean	0.127557	20.249817
std	3.066631	4.270510
min	-2.994037	1.000000
25%	-0.046665	18.000000
50%	-0.010701	20.000000
75%	0.029811	23.000000
max	109.852908	36.000000

Figure 11: Summary statistics of applying MACD on all tickers in HKEX from the given time span.

Figure 11 shows that the average portfolio return using MACD crossover is about 12.8%, and the standard deviation is around 3.1. At the same time, the mean number of trading signals is about 20.

Scenario 3: Evaluate the best performance given a ticker symbol and a fixed period.

- Dependent variables: Sharpe ratio, Number of trades
- Independent variables: Ticker=0005.HK, Strategy=MACD crossover, Time span=2017-01-01 to 2019-01-01



Figure 12: Heat map illustrates the number of trading signals under different strategies using 0005.HK from 2017 to 2019

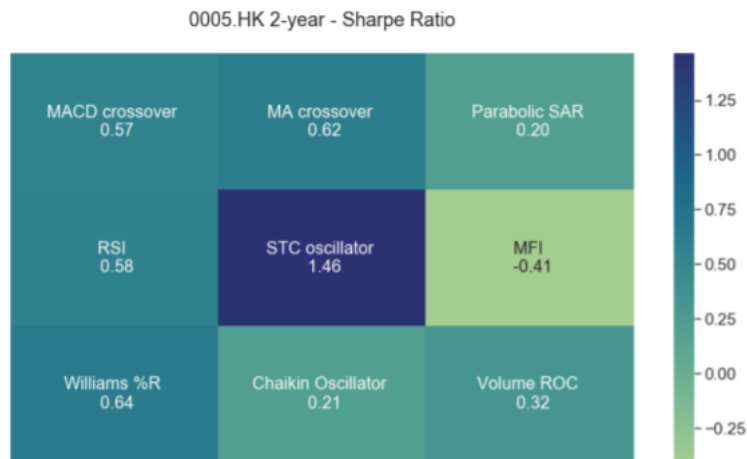


Figure 13: Heat map illustrates the performance of Sharpe ratio under different strategies using 0005.HK from 2017 to 2019.

Figure 12 shows that the Stochastic oscillator (STC oscillator) has the highest Sharpe ratio, which has the best performance among other indicators. Also, Figure 13 reveals a positive relationship between the number of trading signals and the Sharpe ratio of the strategy.

Scenario 4: Examine the correlation between two strategies given a ticker symbol.

- Dependent variables: Sharpe ratio in different periods
- Independent variables: Ticker=0005.HK, Strategy= {MACD crossover, MA crossover, Parabolic SAR, RSI, STC oscillator, MFI, Williams %R, Chaikin Oscillator, Volume ROC}, Time span=2017-01-01 to 2019-01-01

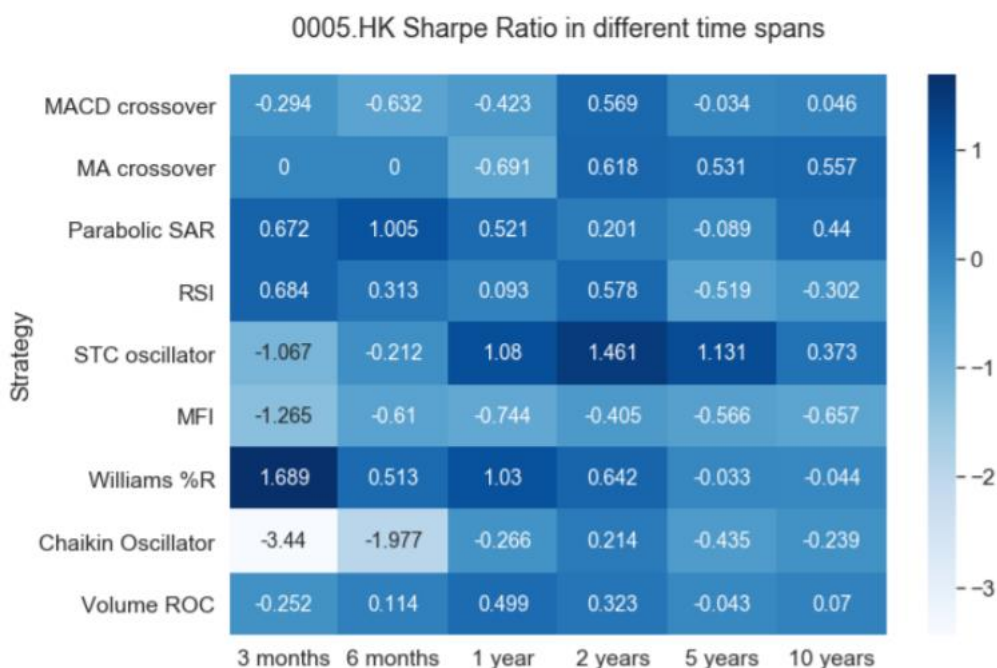


Figure 14.: Heat map that illustrates the performance of Sharpe ratio for different strategies from the given time span using 0005.HK

From Figure 14, we can make the following hypothesis:

- Trend indicators (MACD crossover, MA crossover, Parabolic SAR) perform better in the short term from 3 months to 1 year.
- Momentum indicators, except for Williams %R, tend to improve as the period increases.
- Volume indicators (Chaikin Oscillator, Volume rate of change) has the worst performance among other indicators.

We have made our hypothesis by testing on ticker 0005.HK, which might not represent the general performance of other tickers. The result serves as an illustration purpose of using a heatmap to evaluate different trading strategies. Hence, the hypothesis might only be true under certain assumptions, and more experiments should be carried out to suit the need of different investment purposes.

3.2.1.2 Fundamental Analysis

This part focus on applying fundamental analysis on stock screening. We have collected the Income statement, Balance sheet and Cashflow financial statement in the database to evaluate the actual performance of each listed companies. Investors could filter out the best-performing stocks using a specific criterion (e.g. using EPS as the evaluation metrics). The entire coding was coded on Jupyter Notebook file ratio-analysis.ipynb in the repository. Algorithm 9 shows the major steps to conduct stock screening.

Algorithm 9: Pseudo code for stock screening

Data: Financial statements for tickers in a stock exchange

Result: filtered_df, A list of filtered stocks desirable for investment

1. Import required libraries
 2. merged_df \leftarrow Merge dataframes of three types of financial statements
 3. ratio_df \leftarrow Compute the financial ratios according to the equations (which are available on the documentation website)
 4. masks \leftarrow Create sceening masks
 5. filtered_df \leftarrow Apply the masks to the dataframe
 6. Find the name and sector of the company by matching with the ticker symbol in the master list
-

Algorithm 9: Pseudo code to carry out stock screening

There are four financial ratios covered, including turnover ratios, financial leverage ratio, profitability ratios and short-term solvency ratios. Table 3 shows the commonly used financial ratios used by investors. Users could configure specific screening masks based on their investment preferences on the financial ratios.

3.2.2 Data Analysis on Macroeconomic data

Economic indicator and transaction data analysis were performed in this section. Economic indicator analysis examines how the economic indicators affect the monthly average house price per saleable area in Hong Kong, while the transaction data analysis examined the relationship between different property features and the individual housing prices of Hong Kong. Both univariate and bivariate methods were carried out in the two analysis.

3.2.2.1 Economic indicator analysis

We have also explored the relationship between the economic indicators and the monthly average house price per saleable area in Hong Kong. It is chosen as the experiment factor as there is no correlation between the average monthly house price with economic indicators and individual prices in Hong Kong. Both univariate and bivariate analysis was carried out to analyze the relationship of each feature. The monthly average house price per saleable area collected from the Centaline Property was joined with the economic indicators by year and month.

In univariate analysis, `Dataframe.describe()` function was called to examine the distribution of the numerical features. A statistics summary of the mean, standard deviation, min, and max value of the data was shown. Additionally, `seaborn.displot()` function was used to visualize the results with histograms. Algorithm 10 demonstrates the code snippet for univariate analysis.

Algorithm 10: Code for the statistical summary

Data: `feature_name`

1. `Print(df[feature_name].describe())`
 2. `Plt.figure(figsize=(10,5))`
 3. `Sns.displot(df[feature_name], axlabel=var)`
-

Algorithm 10: Code for carrying out the univariate analysis.

We then move on to study the correlations between features in bivariate analysis. The relationship between the two features was visualized using a scatter plot and regression line. Figure 15 shows an example of a scatter plot with a regression line, and there is a negative slope in the regression line as shown in Figure. This indicates that the seasonally adjusted unemployment rate has a negative correlation with the monthly average price per saleable area.

A correlation matrix was computed and visualize using the `pandas.DataFrame.corr()` and `seaborn.heatmap()` function. Table 2 shows the correlation coefficient of the economic indicators and the monthly average house price per saleable area in Hong Kong. From the result, there is a positive correlation between indicators including GDPs per capita, GDP, composite consumer price index, population, year, imports, month, and total exports) with the monthly average house price per saleable area in Hong Kong. However, both the seasonally and the non-seasonally adjusted unemployment rate negatively correlate with the monthly average house price saleable area in Hong Kong.

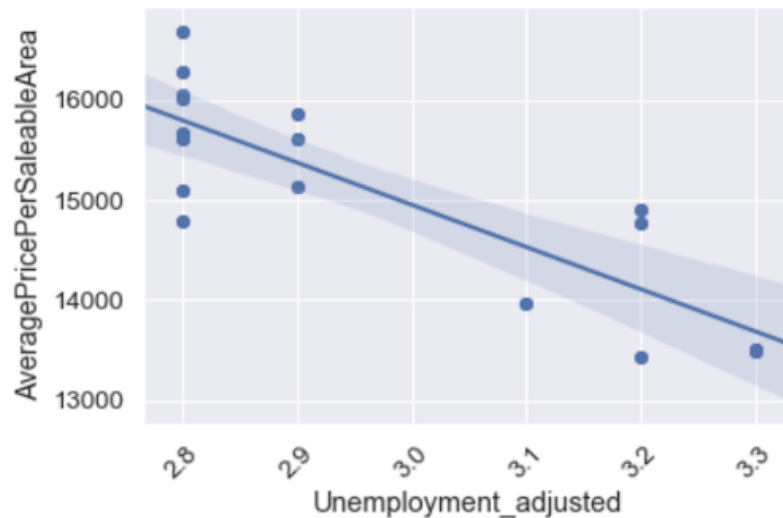


Figure 15: Regression output of Average Price saleable area with the adjusted unemployment.

Feature	Correlation coefficient
GDP per capita	0.82
GDP	0.70
Composite consumer price index	0.69
Population	0.68
Year	0.68
Imports	0.68
Month	0.38
Total exports	0.37
Unemployment rate (not seasonally adjusted)	-0.65
Unemployment rate	-0.84

Table 2: Table shows the correlation coefficient between the economic indicators and the monthly average house price per saleable area in Hong Kong.

3.3.2.2 Transaction data analysis

We also examined the distribution of Hong Kong's house price in the univariate analysis. From the experiment, the mean Hong Kong housing price has a mean of 9 million HSD and a standard deviation of 13 million HKD. On top of that, the skewness and kurtosis were 26.9 and 1526.4, respectively. The experiments show that Hong Kong's housing price is skewed positively to a high degree.

Before carrying out the bivariate analysis, outliers were removed by using standard deviation. In bivariate analysis, the correlation coefficient between the property features and the house price was computed, and the top 7 features with the highest correlation were selected. Figure 16 shows the correlation heatmap of the top 7 features.

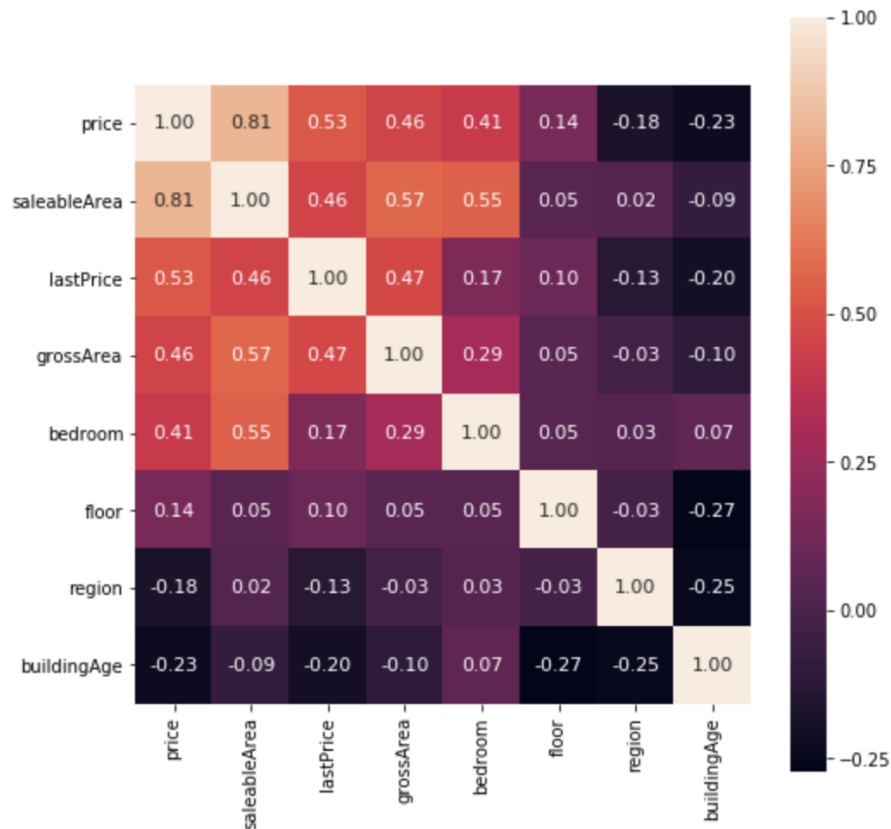


Figure 16: Correlation heatmap showing the correlation coefficient in the property transaction data.

From Figure 16, the housing price in Hong Kong following results were obtained:

- a strong positive correlation with the saleable area
- a moderate positive correlation with the number of bedrooms, gross area, and last transaction price
- a weak positive correlation with the floor, region
- a weak negative correlation with building age

3.3 Machine learning predictions

Different machine learning algorithms were performed on the property price prediction and the sentiment analysis.

3.3.1 Bankruptcy prediction

We used machine learning to use fundamentals data to predict the bankruptcy forecast of a stock company. The following variables were chosen as the input features for the machine learning model based on Simple Analysis of Failure (SAF2002) model (Shirata,2003), Altman (2013) and Beaver (1996)

X1 = working capital ÷ total assets

X2 = retained earnings ÷ total assets

X3 = Earnings before interest and taxes (EBIT) ÷ total assets

X4 = total equity (book) ÷ total assets

X5 = net income ÷ total assets

X6 = total liabilities ÷ total assets

X7 = cash flow from operation ÷ total liabilities

Category	Ratio(s)
Short-term solvency ratios	<ul style="list-style-type: none"> • Current ratio • Quick ratio • Cash ratio • Networking capital to current liabilities
Turnover ratios	<ul style="list-style-type: none"> • Average collection period • Inventory turnover ratios • Receivable turnover • Fixed asset turnover • Total asset turnover
Financial leverage ratios	<ul style="list-style-type: none"> • Total debt ratio • Debt to equity • Equity ratio • Long-term debt ratio • Times interest earned ratio
Profitability ratios	<ul style="list-style-type: none"> • Gross profit margin • Net profit margin • Return on assets (ROA) • Return on equity (ROE) • Earning per share (EPS)

Table 3: List of commonly used financial ratios in fundamental analysis.

The machine learning model is pre-trained with a well-labelled dataset collected from the UCLA School of LAW. The datasets contain more than 200 listed companies in the USA, the label of 0 indicate that the company has gone bankrupt within three years, while the label of 1 shows the company has survived.

In the training process, the dataset split into train and validation set of 7:3. Different typical machine learning models have experimented including Support Vector Machine (SVM),

Decision Tree, Random Forest, and K-nearest neighbours. We have adopted accuracy to be the evaluation metrics. Table 4 shows the summary result of the experiments with the training dataset. We can observe that Random Forest has the highest accuracy score in predicting bankrupt in one year. While considering the long-term prediction, both SVM and Random Forest have the highest accuracy score of 65% in predicting the three-year bankruptcy possibility of a company.

Model	1-year	3-year
Support Vector Machine (SVM)	73%	65%
Decision Tree	69%	54%
Random Forest	88%	65%
K-Nearest Neighbour (KNN)	77%	50%

Table 4: Accuracy score of different machine learning models in predicting 1-year and 3-year bankruptcy

3.3.2 Property Price Prediction

- Output variable: Housing price
- Input variable: Top 7 features from the analysis
- Data: The price data split with the ratio of 8:2 to be used as train and test data
- Data Processing: Log transformation was used to normalize the highly skewed price data to reduce the dynamic range of Hong Kong's property prices.

Four different machine learning models, including XGBoost, Lasso, Random Forest, and Linear Regression, were carried out for the prediction. Root mean square (RMSLE) was chosen as the evaluation metrics, as it prevents penalizing significant differences between predicted and actual price. Table 5 shows the RMSLE value of different models.

Model	RMSLE	
	Train	Test
XGBoost	0.1608	0.1645
Lasso	0.2640	0.2652
Random Forest	0.3077	0.3071
Linear Regression	0.2630	0.2643

Table 5: Table with the RMSLE value of different models

From the output in Table 5, XGBoost has the best performance as it uses a more accurate implementation of gradient boosting algorithms and optimized regularisation. However, there is an overfitting issue in training the data. Figure 17 shows the graph of the actual and predicted property price for XGBoost, and we can observe that a larger number of training data produce a smaller error.

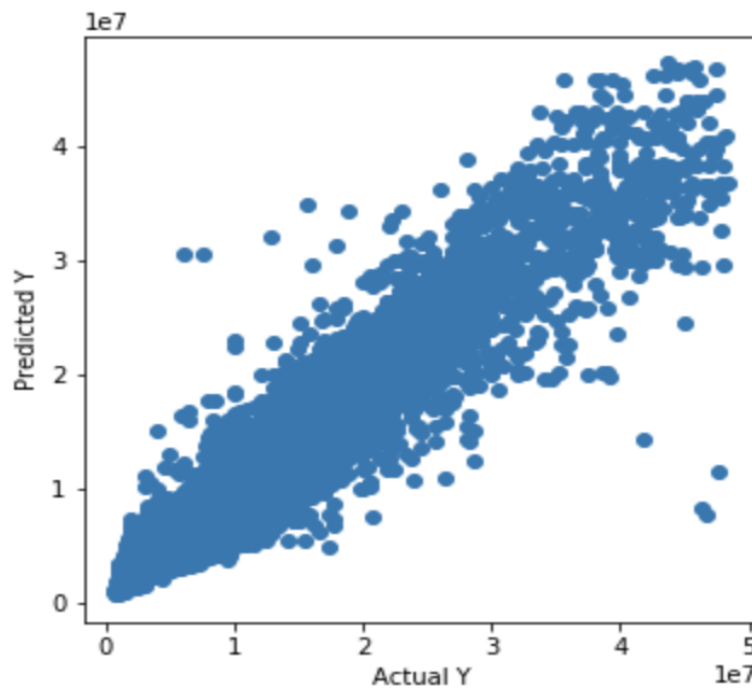


Figure 17: The graph of predicted and actual house price using XGBoost model.

3.3.3 Sentimental analysis using Vader library

VADER is unsupervised learning. We experimented with the VADER model, which is a simple rule-based model for sentiment analysis of social media text. Results show that it achieves high accuracy on a well-labelled dataset. The VADER Compound score (Hutto and Gilbert, 2014) is a metric that sums up all the lexicon ratings (for each tweet), which have been normalized between -1 (most extreme negative) and +1 (most extreme positive). Figure 18

shows the Vader label after running Vader library on 5000 positive tweets and 5000 negative tweets from the NTLK library. We can see that most negative tweets have a Vader label score below 0 and positive tweets have a Vader label score above 0.

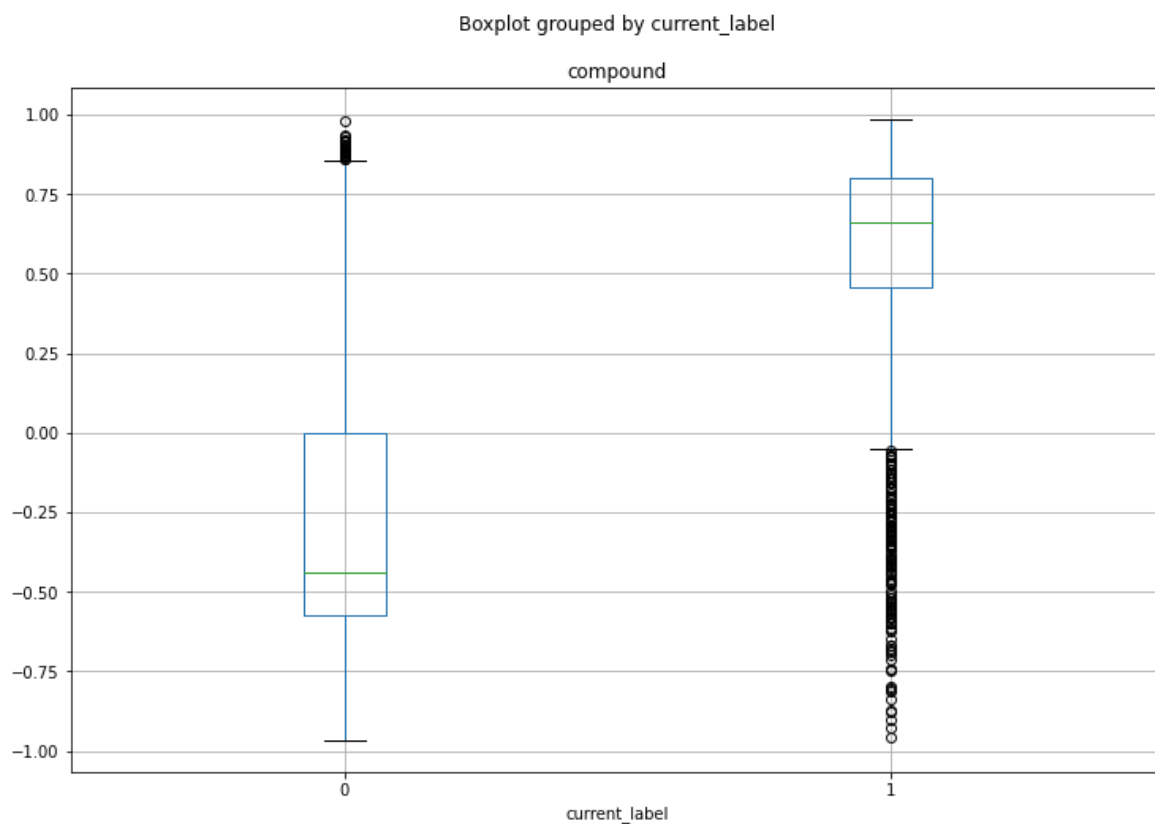


Figure 18: Figure shows the Vader label of tweets samples from NTLK library.

Thus, the VADER model was employed to label the dataset of tweets and news headlines. In the data repository, with the steps shown in Figure 22. The sentiment labels generated from the VADER Compound score using Algorithm 11.

Algorithm 11: Retrieve the Vader score

Data: news or twitter datasets of a ticker

Result: a csv file contains the pandas dataframe of Vader sentiment score from 0 to 2

1. Go through the [SentimentIntensityAnalyzer \(\)](#) and append a compound Vader score to every news row
 2. Group the daily mean compound Vader score by dates
 3. Convert the Vader score using a threshold to a sentiment label
 4. Output the csv file containing the Vader label of a ticker in different dates
-

Algorithm 11: Pseudo code to retrieve the Vader score

3.3.4 Sentimental analysis using Text blob library.

TextBlob is a python library that provides a consistent API for diving into common NLP tasks like sentiment analysis from their official website. It will output a polarity score from -1 to 1. In the sentiment analysis of news text from aastock.com, we have accumulated the daily sentiment score of the text and classify it into different sentiments.

Algorithm 12: Retrieve the TextBlob score

Data: news or twitter datasets of a ticker

Result: a csv file contains the pandas dataframe of TextBlob sentiment score from 0 to 2

1. Import TextBlob function from TextBlob module
 2. Go through the `TextBlob ()` function and append the polarity score to every news row
 3. Group the daily mean TextBlob score by dates
 4. Convert the TextBlob score using a threshold to a sentiment label
 5. Output the csv file containing the TextBlob label
-

Algorithm 12: Pseudo code to retrieve the TextBlob score

Algorithm 13 shows the threshold function for both the Vader and Textblob, to convert its corresponding compound score into sentiment label:

Algorithm 13: Threshold function

Positive sentiment (= 2): compound score > 0.01

Negative sentiment (= 0): compound score < -0.01

Neutral sentiment (= 1): $-0.01 \geq \text{compound score} \leq 0.01$

Algorithm 13: Pseudo code for the threshold function

A threshold of 0.01 was set because the average daily percentage move in the stock market is between -1% and +1 % reference to the S&P 500. (Financial Samurai, 2021)

3.3.5 Sentimental analysis using Loughran-McDonald Financial Dictionary 2018

Loughran and McDonald manually construct this dictionary. It contains seven sentiment dimensions: positive, negative, uncertainty, litigious, weak modal, strong modal, and constraining. We only used positive, negative, and uncertainty dimensions to transfer news text into the Loughran label (positive, negative or neutral). (Bastos, 2020) We have chosen this finance domain specific sentiment dictionary model, as it brings better prediction performance than other dictionary. (Xiaodong Li*, Pangjing Wu, Wenpeng Wang, 2019)

Algorithm 14: retrieve the Loughran label

Data: news or twitter datasets of a ticker

Result: a csv file contains the pandas dataframe of Loughran sentiment score from 0 to 2

1. Import TextBlob function from TextBlob module
 2. Call the `build_dict ()` function to load in the Loughran master dictionary, stop words and return a list of positive words, negative words and stop list
 3. Call the `process_news_loughran ()` function to tokenize the word in each news and remove the stop words
 4. Go through the threshold function to convert the result into Loughran label
 5. Output the csv file containing the Loughran label
-

Algorithm 14: Pseudo code to retrieve the Loughran label

Algorithm 15: Threshold function

Positive sentiment (= 2): $\text{len}(\text{words_new_pos}) > \text{len}(\text{words_new_neg})$

Negative sentiment (= 0): $\text{len}(\text{words_new_pos}) < \text{len}(\text{words_new_neg})$

Neutral sentiment (= 1): $\text{len}(\text{words_new_pos}) == \text{len}(\text{words_new_neg})$

Algorithm 15: Pseudo code for the threshold function in Loughran function

4. Experiments

Apart from performing different data analysis and machine learning prediction of the three market aspects, we have carried out experiments incorporating different market indicators to form a multi-feature machine learning input to predict stock price.

4.1 Datasets

We used stock data from HKEx to examine the performance of the stock price predictions after going through data analysis from three market perspectives. The datasets covered in this experiment include data from January 2017 to March 2021.

- Technical indicator: MACD crossover
- Sentiment indicator: TextBlob signal, Vader signal
- macroeconomic indicators: GDP, Monthly average house price and the Unemployment rate

There are thousands of stocks traded in HKEx, but not all of them are actively traded in the Hong Kong market, and they tend to have less trading volume and market capital. Our experiment mainly focuses on the constituents of the Hang Seng index (HSI), as these stocks

tend to have significant market media coverage and trading volume comparing with other stocks in the market. To form a more representative approach, we have selected three representative stocks in each sector of HIS. The detailed statistics were shown in Table 6 and 7.

	ticker	firm_name	industry
0	669	TECHTRONIC IND	Consumer Discretionary
1	175	GEELY AUTO	Consumer Discretionary
2	1211	BYD COMPANY	Consumer Discretionary
3	2319	MENGNU DAIRY	Consumer Staples
4	168	CHINA RES BEER	Consumer Staples
5	1	CKH HOLDINGS	Conglomerates
6	4	WHARF HOLDINGS	Conglomerates
7	19	SWIRE PACIFIC A	Conglomerates
8	700	TENCENT	Information Technology
9	9988	BABA-SW	Information Technology
10	823	LINK REIT	Properties & Construction
11	16	SHK PPT	Properties & Construction
12	1299	AIA	Financials
13	5	HSBC HOLDINGS	Financials
14	939	CCB	Financials

Table 6: Table shows the list of selected tickers - part 1

	ticker	firm_name	industry
15	2	CLP HOLDINGS	Utilities
16	3	HK & CHINA GAS	Utilities
17	2688	ENN ENERGY	Utilities
18	941	CHINA MOBILE	Telecommunications
19	762	CHINA UNICOM	Telecommunications
20	968	XINYI SOLAR	Industrials
21	868	XINYI GLASS	Industrials
22	2382	SUNNY OPTICAL	Industrials
23	2899	ZIJIN MINING	Materials
24	1818	ZHAOJIN MINING	Materials
25	2689	ND PAPER	Materials
26	883	CNOOC	Energy
27	386	SINOPEC CORP	Energy
28	857	PETROCHINA	Energy

Table 7: Table shows the list of selected tickers - part 2

4.1.1 Datasets from Sentiment factor

There are 31552 rows of data in the sentiment datasets from January 2017 to March 2021.

- `df.groupby(by='vader_label').size()` function was used to find out that in this 5-year, 5162 entry has Vader label of 0, 2751 entries have Vader label of 1, and 23639 entries has Vader label of 2.
- `df.groupby(by='textblob_label').size()` function was used to find out that in this 5-year, 5092 entry has TextBlob label of 0, 6652 entries have TextBlob label of 1, and 19808 entries has TextBlob label of 2.

	dates	vader_label	textblob_label	ticker	firm_name	industry
0	2017-01-02	0.0	0.0	669	TECHTRONIC IND	Consumer Discretionary
1	2017-01-03	2.0	2.0	669	TECHTRONIC IND	Consumer Discretionary
2	2017-01-04	2.0	2.0	669	TECHTRONIC IND	Consumer Discretionary
3	2017-01-05	0.0	2.0	669	TECHTRONIC IND	Consumer Discretionary
4	2017-01-06	2.0	2.0	669	TECHTRONIC IND	Consumer Discretionary
...
31547	2021-02-25	2.0	2.0	857	PETROCHINA	Energy
31548	2021-02-26	2.0	2.0	857	PETROCHINA	Energy
31549	2021-03-01	1.0	1.0	857	PETROCHINA	Energy
31550	2021-03-02	2.0	2.0	857	PETROCHINA	Energy
31551	2021-03-03	2.0	2.0	857	PETROCHINA	Energy

31552 rows × 6 columns

Table 8: Table shows the Vader and TextBlob sentiment label for daily news of selected tickers from January 2017 to March 2021

4.1.2 Datasets from Technical factor

Table 9 shows the list of data from the microeconomic database. We used the closing price of each ticker from January 2017 to March 2021 for different experiment setting. The closing price were compared with the predicted price to arrive at a signal indicator. While Table 9 shows the mean score of MACD, returns and signal in each industry, it is observed that each industry is having a similar signal score from the MACD crossover strategies.

	Date	Close	MACD	Signal line	returns	signal	ticker	firm_name	industry
0	2017-01-03	27.115299	-0.340230	-0.282324	0.000000	0.0	669	TECHTRONIC IND	Consumer Discretionary
1	2017-01-04	26.407944	-0.340996	-0.294059	0.000000	0.0	669	TECHTRONIC IND	Consumer Discretionary
2	2017-01-05	26.879515	-0.300092	-0.295265	0.000000	0.0	669	TECHTRONIC IND	Consumer Discretionary
3	2017-01-06	26.266474	-0.313528	-0.298918	0.000000	0.0	669	TECHTRONIC IND	Consumer Discretionary
4	2017-01-09	26.172157	-0.328006	-0.304736	0.000000	0.0	669	TECHTRONIC IND	Consumer Discretionary
...
30851	2021-02-25	2.880000	0.102874	0.060587	0.000069	1.0	857	SINOPEC CORP	Energy
30852	2021-02-26	2.780000	0.101515	0.068773	-0.000099	1.0	857	SINOPEC CORP	Energy
30853	2021-03-01	2.810000	0.101686	0.075356	0.000030	1.0	857	SINOPEC CORP	Energy
30854	2021-03-02	2.720000	0.093482	0.078981	-0.000089	1.0	857	SINOPEC CORP	Energy
30855	2021-03-03	2.800000	0.092371	0.081659	0.000079	1.0	857	SINOPEC CORP	Energy

30856 rows × 9 columns

Table 9: Table shows the fundamental data for daily information of selected tickers from January 2017 to March 2021.

	MACD	returns	signal
industry			
Information Technology	1.876696	1.590718e-04	0.526770
Properties & Construction	0.279655	9.598261e-06	0.501361
Telecommunications	0.044132	6.646339e-06	0.516788
Conglomerates	-0.082177	6.966165e-06	0.543255
Consumer Discretionary	0.720008	4.577083e-05	0.515426
Consumer Staples	0.277927	5.886660e-06	0.481397
Energy	-0.009926	-5.670379e-07	0.490926
Financials	0.001530	-5.152576e-06	0.484876
Industrials	0.441671	5.056694e-05	0.527223
Materials	0.022384	2.475248e-06	0.516334
Utilities	0.161317	9.684852e-06	0.507260

Table 10: Table shows the mean value of MACD, returns and signal in each industry

4.1.3 Datasets from macroeconomic factor

Table 11 shows the input data from the macroeconomic data. We used the monthly GDP, unemployment rate, and property price from January 2017 to March 2021 as the input feature from the macroeconomic aspect in the LSTM-multi-feature model. While Table 12 shows the summary statistic of macroeconomic data after normalize it into the range of -1 to 1. The mean score of GDP and Property price is quite high closing to 0.9 with a low standard deviation, and the mean unemployment rate is quite high with over 60% and a standard deviation of 0.2.

	Date	GDP	Unemployment rate	Property price	ticker	firm_name	industry
0	2017-01-03	0.837848	0.515625	0.743503	669	TECHTRONIC IND	Consumer Discretionary
1	2017-01-04	0.837848	0.515625	0.743503	669	TECHTRONIC IND	Consumer Discretionary
2	2017-01-05	0.837848	0.515625	0.743503	669	TECHTRONIC IND	Consumer Discretionary
3	2017-01-06	0.837848	0.515625	0.743503	669	TECHTRONIC IND	Consumer Discretionary
4	2017-01-09	0.837848	0.515625	0.743503	669	TECHTRONIC IND	Consumer Discretionary
...
30851	2021-02-25	0.845069	1.000000	0.945556	857	SINOPEC CORP	Energy
30852	2021-02-26	0.845069	1.000000	0.945556	857	SINOPEC CORP	Energy
30853	2021-03-01	0.845069	1.000000	0.945556	857	SINOPEC CORP	Energy
30854	2021-03-02	0.845069	1.000000	0.945556	857	SINOPEC CORP	Energy
30855	2021-03-03	0.845069	1.000000	0.945556	857	SINOPEC CORP	Energy

30856 rows × 7 columns

Table 11: Table shows different indicator under macroeconomic aspect.

	GDP	Unemployment rate	Property price
count	30856.000000	30856.000000	30856.000000
mean	0.903488	0.606000	0.914839
std	0.054334	0.226532	0.061186
min	0.833680	0.437500	0.743503
25%	0.845069	0.437500	0.871557
50%	0.901453	0.484375	0.940748
75%	0.954496	0.921875	0.958290
max	1.000000	1.000000	1.000000

Table 12: Table shows the summary statistics in the macroeconomic data

4.2 Baseline Setup

The baseline model does not involve machine learning training. Instead, we combined different technical indicators as layers of filter to screen out the proper signal. The steps were carried in the following description:

1. Apply one of the technical strategies (MACD crossover in the current experiment) to get the technical signal
2. Adjust the bias in signals with macroeconomic data with {unemployment rate, GDP and Property price}
 - finding out the correlation of price using the `dataframe.corr()` function into `s_gdp`, `s_unemploy` and `s_property` variables
 - Normalize the macroeconomic factor from a range of -1 to 1 and store into corresponding names of `signals['GDP']`, `signals['Unemployment rate']` and `signals['Property price']`
 - Calculate the adjusting signal by using the following equations show in Algorithm 16

Algorithm 16: updating signals after macroeconomic filter

```
signals['macro_factor'] = s_gdp * signals['GDP'] +
                        s_unemploy * signals['Unemployment rate'] +
                        s_property * signals['Property price']
signals['signal'] = signals['signal'] + signals['macro_factor']
```

Algorithm 16: Code for updating signals after the macroeconomic filter

3. Filter out data that contrast with the sentiment label using the following procedures:
 - Set signal to 0 where there is buy signal from upper stream and a negative sentiment label
 - Set signal to 0 where there is sell signal from upper stream and positive sentiment label
4. Carry out different evaluation strategy on the output using Portfolio return , Sharpe ratio, Max drawdown and Compound annual growth rate.

4.3 LSTM setup

We have adopted LSTM as our machine learning model. Long short-term memory network (LSTM) is one type of RNN and has the memory ability to improve prediction performances. (Bastos, 2020)The details of the model were explained in the following subsections. Data is first ensuring to the floating variables, and all the input features are normalized using the MinMaxScaler from -1 to 1. Adam optimizer was implemented on the single and multi-feature LSTM model. The learning rate of 0.01 was used for all the LSTM model.

4.3.1 LSTM single feature model

Two-layer model and a final fully connected model as used to retrieve the predicted closing price. The batch size of the LSTM neural network was set to 32, and the number of epochs was set to 100. All the code could be found in [../integrated-strategy/LSTM-train_price-only_wrapper.py](#).

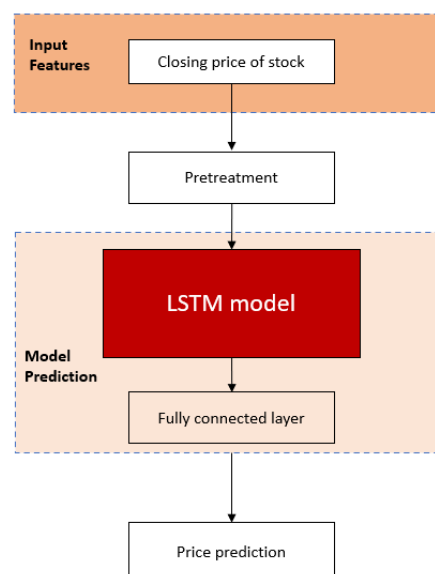


Figure 19: High level workflow of LSTM single feature model

4.3.2 LSTM multiple feature model

Four-layer model and a final fully connected model as used to retrieve the predicted closing price. The experiment set up with a batch size of 72 and number of epochs is 100. Figure 20 shows the high level summary of the workflow carried out in the model. All the code could be found in [../integrated-strategy/LSTM-train_wrapper.py](#).

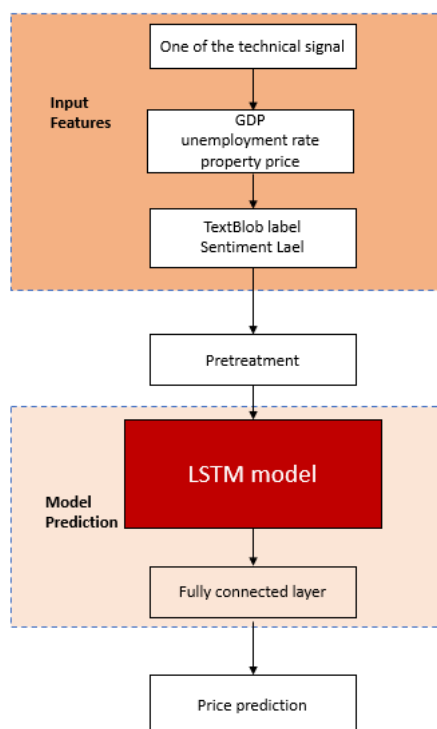


Figure 20: High level workflow of LSTM multi-feature workflow

5 Results

This section discusses the results from the above experiment setup. 28 representative tickers listed in section 4.1 from the HKEx market was used as an example to show different performance during the prediction period. The performance graph under different experiment setting is shown below.

5.1 Baseline Model

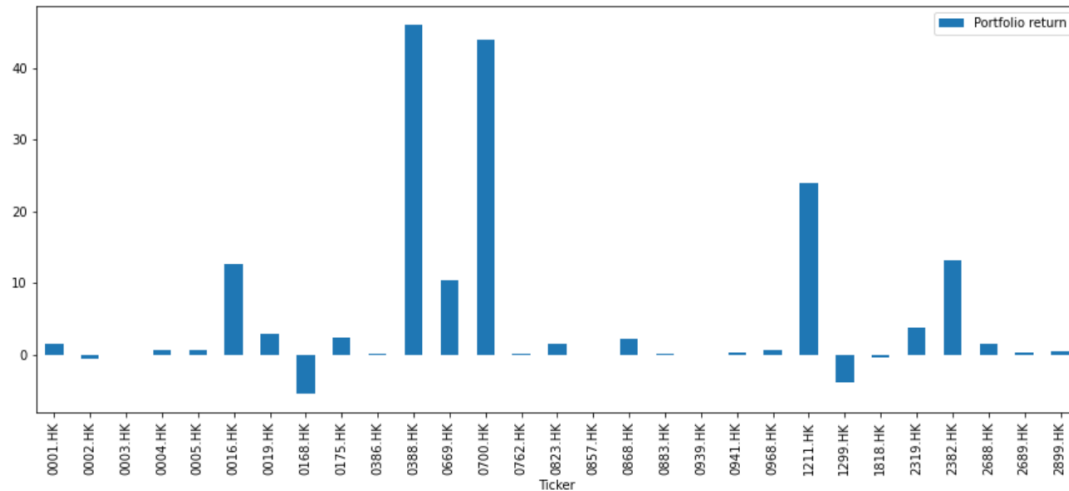


Figure 21: Bar graph shows the portfolio return from 2020-06-10 to 2021-03-03.

	Portfolio return	Sharpe ratio	CAGR	Trade signal number
count	29.000000	29.000000	29.000000	29.000000
mean	5.458059	0.677288	0.051315	7.724138
std	12.372515	1.019278	0.116123	2.051072
min	-5.505000	-1.817451	-0.052230	4.000000
25%	0.089169	0.149553	0.000845	6.000000
50%	0.589196	0.867680	0.005581	8.000000
75%	2.880096	1.372951	0.027265	9.000000
max	46.067200	2.703244	0.431832	12.000000

Table 13: Table shows the summary statistics under the baseline model from 2020-06-10 to 2021-03-03.

We can see from Figure 21 that 0388.HK and 0700.HK perform exceptionally well under the baseline model of more than 40% return. While 0168.HK had a negative return of closing to -5% return. The average performance is still quite pleasing, with a 5.5% cumulative return. It also has a mean and max Sharpe ratio of 0.7 and 2.7 respectively. Which shows that it is making a positive risk-adjusted return.

5.2 LSTM single feature price results

- Input feature: Stock's closing price
- Output variables: Predicted closing price.

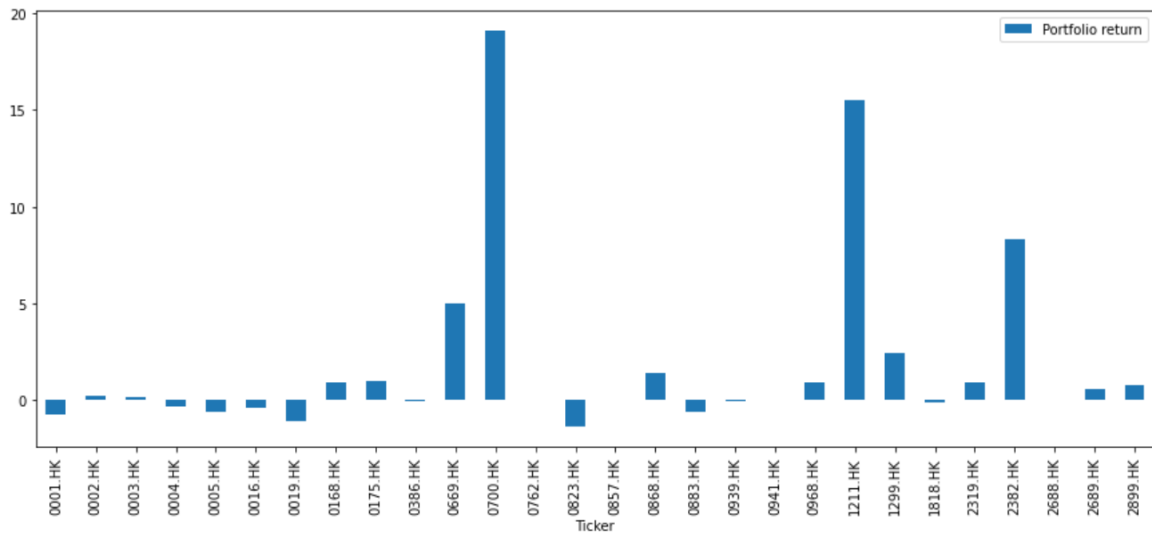


Figure 22: Bar graph shows the portfolio return from 2020-06-10 to 2021-03-03 under LSTM single-feature model

	Portfolio return	Sharpe ratio	CAGR	Trade signal number
count	28.000000	28.000000	28.000000	28.000000
mean	1.837218	0.266442	0.017345	4.142857
std	4.808892	1.323869	0.045374	5.509011
min	-1.399424	-3.018249	-0.013263	0.000000
25%	-0.189616	-0.736502	-0.001796	0.000000
50%	0.061424	0.142675	0.000582	2.000000
75%	0.939271	1.345665	0.008896	4.250000
max	19.139814	2.600999	0.180467	19.000000

Table 14: Table shows the summary statistics under the LSTM single-feature model from 2020-06-10 to 2021-03-03.

We can see from Figure 22 and Table 14 that 0700.HK and 1211.HK are the top 2 best performing stock with more than 15% return. While the mean performance is about 1.83% portfolio return. The standard deviation is lesser compared with the LSTM-multiple feature model which shows the portfolio return is less disperse. In terms of the sharpe ratio, the mean Sharpe ratio is 0.3% and the max is 2.6%. This shows that this trading strategies still still making a positive risk-adjusted returns.

5.3 LSTM multiple feature absolute price results

- Input feature: {Stock's closing price, one of the technical indicators, TextBlob signal, Vader signal, GDP, Monthly average house price}
- Output variables: Predicted closing price.

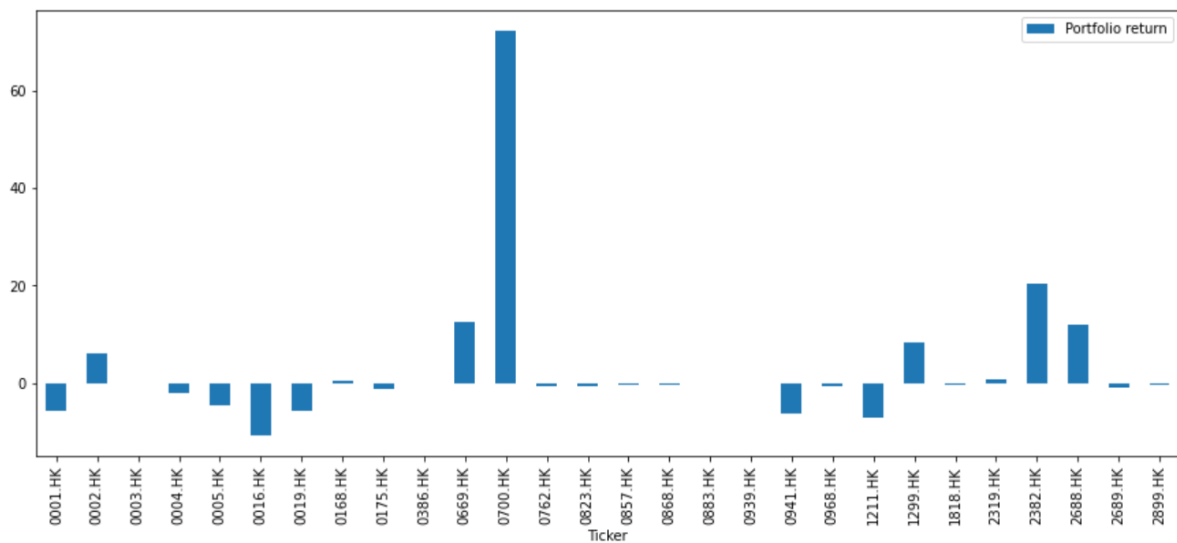


Figure 23: Bar graph shows the portfolio return from 2020-06-10 to 2021-03-03 under LSTM multi-feature model absolute price result.

	Portfolio return	Sharpe ratio	CAGR	Trade signal number
count	28.000000	28.000000	28.000000	28.000000
mean	3.009612	0.016202	0.003816	15.535714
std	15.028250	0.167665	0.020540	11.808604
min	-10.834689	-0.171460	-0.018833	0.000000
25%	-1.460750	-0.105786	-0.002437	5.000000
50%	-0.382835	-0.028607	-0.000636	12.500000
75%	0.507402	0.081159	0.000839	25.500000
max	72.300000	0.539403	0.094394	37.000000

Table 15: Table shows the summary statistics under the LSTM multi-feature model from 2020-06-10 to 2021-03-03.

We can see from Figure 27 and Table 15 that 0700.HK has a return of more than 60%. There are also 3 stocks with a performance return of more than 10% return. There are 7 stocks with a negative return. In which, 0016.HK has the worst performance of -10.83%. In terms of standard deviation, it is around 15, which is quite high showing that the performance of each ticker is quite disperse and the mean performance could not represent the whole performance. Also for the sharpe ratio, it is accounting for the excess return after deducting the risk-free rate. The max and mean value of Sharpe ratio is 0.17 and 0.5 respectively. This shows that this trading strategies still making a positive risk-adjusted returns.

5.4 Conclusion on the model exploration

After exploring various strategy using the machine learning approach and the simple baseline model. Interestingly, the baseline model has the best average performance with a mean performance score of 5.5%. Among all other stocks, 0700.HK tends to achieve the best performance in different approach except for LSTM multi-feature trend prediction for over 60% portfolio return.

We tried to compare our results with the annual percentage change of the Hang Seng Composite Index to benchmark our trading strategies. Table 16 shows the list of annual percentage change of the Hang Seng index from 2017 to 2021. HSI has a -3.40% change in 2020 and a 5.7% change in current 2021. Our baseline model is comparable with the current annual change of HIS with a portfolio return of 5.5% from mid-2020 to early 2021.

Hang Seng Composite Index - Historical Annual Data

Year	Average Closing Price	Year Open	Year High	Year Low	Year Close	Annual % Change
2021	29,047.80	27,472.81	31,084.94	27,472.81	28,793.14	5.74%
2020	25,282.77	28,543.52	29,056.42	21,696.13	27,231.13	-3.40%
2019	27,581.81	25,130.35	30,157.49	25,064.36	28,189.75	9.07%
2018	28,850.94	30,515.31	33,154.12	24,585.53	25,845.70	-13.61%
2017	26,190.41	22,150.40	30,003.49	22,134.47	29,919.15	35.99%

Table 16: Table taken from macro-trends.net showing the list of Historical Annual Data from 2017 to 2021

5.5 Documentation of the website

In response to the project's education motives, we have documented all the code implementations in the following website hosted on Readtheocs (<https://algo-trading.readthedocs.io/en/latest/>). The website was built with Sphinx, which is a documentation generator designed for writing technical documentation.

The tutorials in the repository mainly consisted of four parts:

Part 1: Intro to Algo Trading

This part introduces the basic concept of algo trading. The typical fundamental and technical analysis applied in real-trading situations and tutorials on some essential data science basics.

Part 2: Core Trading Strategies

This part includes a more profound introduction in both Technical analysis and Fundamental analysis. Under the technical analysis, chart and technical analysis were covered, including some explanation of financial ratios and tutorials on ratio analysis essential for stock screening. Lastly, we acknowledge that knowing different evaluation metrics is very important to evaluate trading strategies' performance. Hence, to serve this purpose, different standard evaluation metrics like Portfolio return, Sharpe ratio, Maximum drawdown compound annual growth rate, and standard deviation were covered.

Part 3: Machine learning predictions

This part mainly introduces how users can utilize machine learning methodologies to train data in different aspects. As an example, bankruptcy predictions by applying microeconomic data and property price prediction with macroeconomic data. Similarly, under the sentiment analysis, it introduces the collection of tweets and on financial headlines and uses different machine learning algorithm to find out the sentiment of the text. Lastly, we also introduce how to put all the data and results together to form a comprehensive trading strategy.

Bonus part:

Paper Trading execution: introduced the basic setup to connect to the interactive broker API and operations like requesting market data, managing orders, and requesting account summary. Julia is for algo trading: Julia is designed for speed, performance, and scalability and is adopted by many prominent financial firms, including BlackRock and State Street as their primary development language. We hope to provide a basic introduction to Julia, introducing some essential data science skills and illustrating how to use Julia to build the strategy, plotting graphs, backtesting, and strategy evaluation.

6 Limitations

6.1 Limitations in web scraping technologies

Most of the web scraping code in this paper relies heavily on the web scraping technology Beautiful soup. The above web scraping code might not work if the source website such as

Yahoo! Finance changes its front-end design and leads to changes in the HTML document, data collection might become invalid. It requires the user to adjust the code to suit the front technology design in such a situation.

6.2 limitation in API rate limit:

There is an API limitation of collecting the historical tweets. For the standard Search API only supports the history of 7 days, any dates outside of this range could not be collected. User must subscribe to a paid plan to retrieve the entire tweets. Only 450 tweets request could collect in 15 minutes. However, more than 4000 tickers are listed in the US, and we try to collect approximately 200 tweets for each ticker daily. It takes a few days to finish the data collection time that was set a few days ago. Hence the database might not be able to present each day's tweets.

6.3 Limitation in the amount of data collected.

Different indicators from the microeconomic, macroeconomic and sentiment market were collected. Data from microeconomic and macroeconomic aspect were clearly defined and readily available on the internet to download the necessary data. As for sentiment data, there is much financial news reporting over the internet and offline. Even though we try to collect from a prominent news agent, it still might not cover every piece of news on the specific stock. The type of media that different investors uses to interpret their sentiment insight can also affect their sentiment. Hence, we hope to provide a general sentiment by scrapping data from popular mainstream websites and where most investors receive their source of data income. There are also chances there is bias in news coverage in popular stocks than less popular stocks. Hence popular stocks from the widespread industry have extensive data and could result in a better price prediction.

7 Conclusion and Future works

In conclusion, we have made one step towards the goal of democratizing education on algorithmic trading and making it more approachable to the public. Currently, the university worldwide does not have any established syllabus that is incorporating the necessary technical and financial knowledge at the same time to provide comprehensive guidelines to algo trading learning. Most courses focus on the theoretical knowledge of algo trading but neglect the

practical practice of applying the strategies to real-life trading. Thus, we highlight the need for a database that consists of data from multiple sources of historical tick data, property prices data, and sentiment analysis data. We have built a pipeline for downloading the necessary data. The database could be updated continuously by running the written code, and users could modify the code to download their necessary data.

In addition to the dataset, the code implementation could also serve as a guide to the user and transfer their trading strategies into code. Furthermore, machine learning has been covered to demonstrate the power of leveraging the data repository to make predictions about company prices and stock trends. All the code was accompanied with clear explanation and equations in the documentation website that we have created.

Moving on, we combined different indicators to form a multi-feature model in predicting stock trends from different data indicators (microeconomic, macroeconomic, sentiment). Results have shown that it has a positive portfolio return with a mean of around 3%. We have also connected to the Interactive Broker's API to connect to the paper trading account to automate the order procedure.

Our coding, coupled with clear explanations in this project, aims to provide easy-to-understand information to the user. We anticipate that this data repository could fulfil our educational and research objectives in supporting the teaching and learning of algo trading in the long run.

References

- Aggarwal, D. (2019). *Will the rapid rise in algo trading leave traditional traders behind?* New Delhi: The Economic Times.
- Altman, E. I. (2013). Predicting financial distress of companies: revisiting the z-score and zeta® models. In *Handbook of research methods and applications in empirical finance*. Edward Elgar Publishing.
- Beaver, W. H. (1966). Financial ratios as predictors of failure. *Journal of accounting research*, pages 71–111.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
- Hendershott, T., Riordan, R., et al. (2009). Algorithmic trading and information. Manuscript, University of California, Berkeley.
- Hutto, C. and Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 8.
- Melin, M. (2017). Aqr: Computers don't replace human stock pickers, they augment them.
- Tse, R. Y. (2001). *Impact of Property Prices on Stock Prices*. Hong Kong: World Scientific Publishing Company.
- Li, Audeliano Wolian, & Bastos, Guilherme Sousa. (2020). Stock Market Forecasting Using Deep Learning and Technical Analysis: A Systematic Review. *IEEE Access*, 8, 185232-185242.
- Schumaker, Robert, & Chen, Hsinchun. (2009). Textual analysis of stock market prediction using breaking financial news. *ACM Transactions on Information Systems*, 27(2), 1-19.
- Shirata, C. (2003). The bankruptcy prediction model—saf 2002 model. Chuo-Keizai Sha.