AN INVESTIGATION ON ALGORITHMIC TRADING PROFITABILITY WITH SENTIMENT ANALYSIS ALGORITHM IN MALAYSIA MARKET

AHMAD FUAD KHALIT

ASIA PACIFIC UNIVERSITY OF TECHNOLOGY AND INNOVATION

(APU)

JANUARY 2021

Abstract

Disruptive technology is common challenges that most company faces in every industry. Financial industry is not exempted from disruptive technology with the used of autonomous trading that is known as Algorithmic Trading. In Malaysia market, the used of Algorithmic trading is only limited to institution trader and retail investor are often left out to utilize this technology due to various reason. Most of retail investor are comfortable to use traditional buy and hold strategy because they are not interested in managing their investment on daily basis while those who interested in algorithmic trading are often terrified by the complexity of building a algorithm model in order for them to participate. In this research, we are trying to provide framework and simple algorithm for retail trader thus democratize the technology to general public in Malaysia. In this research, we presented 6 candidate model to be used and one of them is our hybrid model SMA + Sentiment Analysis. While previous researcher only used either simple moving average or sentiment analysis individually and applied it directly to the market which expose their model limitation when the market behave erratically especially when market volatility is high, our model increase the robustness of both model with the technique we applied. When building all our model, we applied various technique to archive optimal result such as data preprocessing, data transformation and feature engineering and hyperparameter optimization. The result after evaluation and backtesting, we found that the best performing model was our hybrid model SMA + Sentiment Analysis. This high-performance result can be explained by acknowledging the limitation of moving average model that only work on past data without considering the future changes that might occur. Then by adding sentiment analysis score and aggregating both moving average and sentiment score proved to increase the performance of the model in our experiment. With the development of this model, we able to prove that even simple model can outperformed more complex model and hope that this finding will open the accessibility of algorithmic trading for retail trader to explore it without being put off by its complexity.

Table of Contents

TABLE OF CONTENTS ..... **i**

LIST OF FIGURES ..... **iv**

LIST OF TABLES ..... **vi**

CHAPTER 1: INTRODUCTION

1.1 Background **1**

1.1.1 Trading Process 1

1.1.2 Pre-Trade Analysis 3

1.1.3 Algorithmic Trading System 3

1.2 Problem Statement **4**

1.3 Research Question **5**

1.4 Aim & Objective **5**

1.5 Scope **6**

1.6 Significant of Research **6**

CHAPTER 2: LITERATURE REVIEW

2.1 Position Trading (Buy/Hold Strategy) **7**

2.2 Sentiment Analysis **8**

2.1 Machine Learning **10**

CHAPTER 3: Research Methodology

3.1 Introduction **11**

3.2 Research Approach **11**

3.2.1 Portfolio Selection 12

3.2.2 Data Collection 12

3.2.2.1 Sentiment Analysis (Phase I) 12

3.2.2.2 Machine Learning (Phase II) 12

3.2.3 Text/Data Processing 13

3.2.3.1 Sentiment Analysis (Phase I) 13

3.2.3.2 Machine Learning (Phase II) 14

3.2.4 Polarity Detection Algorithm 14

3.2.5 Data Exploration 15

3.2.6 News Polarity Score 15

3.2.7 Machine Learning Model 15

3.2.8 Evaluation and Back testing 16
3.3 Summary **17**
CHAPTER 4: RESEARCH PLAN
Research Plan **18**
CHAPTER 5: Methodology
5.1 Introduction **19**
5.1.1 Research Gap 19
5.1.2 Research Question 19
5.1.3 Research Objective 20
5.2 Data Collection **21**
5.2.1 Sentiment Analysis 21
5.2.2 Machine Learning 22
5.3 Data Preprocessing **24**
5.3.1 Dataset Explanation 24
5.3.2 Preprocessing 25
5.3.3 Feature Selection 26
5.4 Data Transformation **27**
5.4.1 Feature Engineering 27
5.5 Data Exploration **29**
5.6 Model Development **31**
5.6.1 Simple Moving Average + Sentiment Analysis 31
5.6.2 Linear Regression 32
5.6.3 Gaussian Naïve Bayes 32
5.6.4 Support Vector Machine 33
5.6.5 Random Forest 33
5.6.6 Multi-Layer Perceptron 33
5.7 Evaluation & Validation **34**
CHAPTER 6: Result and Analysis
6.1 Introduction **37**
6.2 Baseline Benchmark **37**
6.3 Phase I Result **39**
6.3.1 Hyperparameter Search & Training the Model 39
6.3.2 Backtesting 41

6.4 Phase II Result **43**
6.4.1 Hyperparameter Search & Training the Model 43
6.4.2 Evaluation 46
6.4.3 Backtesting 48
6.4.3.1 Comparation Between Top 3 Model Performance 49
6.5 Overall Result **52**
6.6 Comparation with Previous Work **53**
6.7 Summary **53**
CHAPTER 7: Discussion and Conclusion
7.1 Introduction **54**
7.2 Discussion and Conclusion **54**
7.3 Important and Contribution Of The Study **56**
7.4 Future Recommendation **56**
REFERENCE 57
APPENDIX A: JUPYTER NOTEBOOK CODE 62
APPENDIX B: ETHICAL APPROVAL OF RESEARCH PROJECT 82
APPENDIX C: LOG SHEET FOR SUPERVISOR SESSION 86
APPENDIX D: TURNITIN SIMILARITY REPORT 90

List Of Figures

Figure 1.1: Order Processing Flow **2**

Figure 2.1: Sentiment Analysis Technique **8**

Figure 3.1: Proposed Research Design **11**

Figure 3.2: Example of Time Series Dataset Extracted From yfinance Library **13**

Figure 3.3: Support Vector Machine **16**

Figure 4.1: Research Plan Timeline **18**

Figure 5.2.1 Web Scraping Code Snippet **18**

Figure 5.2.2 Data collection code **18**

Figure 5.3.1 Web Scraping Data before preprocessing **18**

Figure 5.3.2 Top Glove dataset from Yahoo Finance **18**

Figure 5.3.3 The HTML Tags & Metadata **18**

Figure 5.3.3.1 Loughran and McDonald Financial Sentiment Dictionary (source: https://www.kovcomp.co.uk/wordstat/Sentiment.html) **18**

Figure 5.3.4 Data extraction and feature selection **18**

Figure 5.4.1 Creating binary direction from data **18**

Figure 5.4.2 Binary Direction Lag Return **18**

Figure 5.5.1 Histogram of Sentiment Analysis Score **18**

Figure 5.5.2 Top Glove Closing Price & Daily Return **18**

Figure 5.6.1 Simple Moving Average Formula **18**

Figure 5.6.2 Gaussian Naïve Bayes Algorithm **18**

Figure 5.6.3 Multi Layer Perceptron with single hidden layer (source: Djuriš et al. 2012) **18**

Figure 6.1 Baseline Model Buy and Hold Model **18**

Figure 6.1.1 Baseline Performance Chart **18**

Figure 6.2 Moving Average of 5, 15 and 200 ..... **18**

Figure 6.2.1 Moving Average after hyperparameter search **18**

Figure 6.2.2 Trading Signal of SMA + Sentiment Analysis **18**

Figure 6.3 Tear Sheet Summary Result **18**

Figure 6.3.1 Backtesting Performance Chart **18**

Figure 6.4 Summary Result All Model **18**

Figure 6.4.1 Performance graph on all model **18**

Figure 6.4.2 Number of Transaction **18**

Figure 6.4.3 Accuracy score for all model **18**

Figure 6.4.4 Precision recall evaluation metrics **18**

Figure 6.4.5 Tear Sheet for Top 3 Model **18**

Figure 6.4.6 MLP Performance Chart **18**

Figure 6.4.7 Random Forest Performance Chart **18**

Figure 6.4.8 SVM Performance chart **18**

List Of Tables

Table 6.1 Evaluation result for SMA Model **2**

Table 6.4.1 Multi-Layer Perceptron Hyperparameter Optimizing **2**

Table 6.4.2 Random Forest Hyperparameter Optimizing **2**

Table 6.5 Overall Performance of the Candidate Model **2**

Table 6.6 Previous Work Comparation **2**

Chapter 1

Introduction

1.1 Background

Since early 2000, most of the world's major stock exchanges have transitioned from traditional trading into computerized electronic trading through limit order thus its create a need of a models that understand these new market. Trading mechanisms and regulation also has evolved to keep up with the changes. With the increase of computation power led a revolution to financial market with trading moving from trader floor into data warehouse servers. Server running a programs to process data and submit orders from trader all over the world in fractions of second and with that it has made financial exchanges become more cost effective with improved spreads, faster execution time and low in brokerage commission (Angel, Harris & Spatt, 2011).

1.1.1 Trading Process

Figure 1.1 show how an order is executed in the market exchange. The process start with analyst insight that lead to decision to trade. The insight then moves to portfolio manager to accessed and approved then portfolio manager will give instruction to buy or sell the securities to trader who job is to decide the best approach to execute the order. Direct Market Access refer to direct connection access to the exchange matching engine that execute the order by pairing buy order with sell order. This access is highly regulated by the regulator and normally granted to license brokerage firm to execute an order.

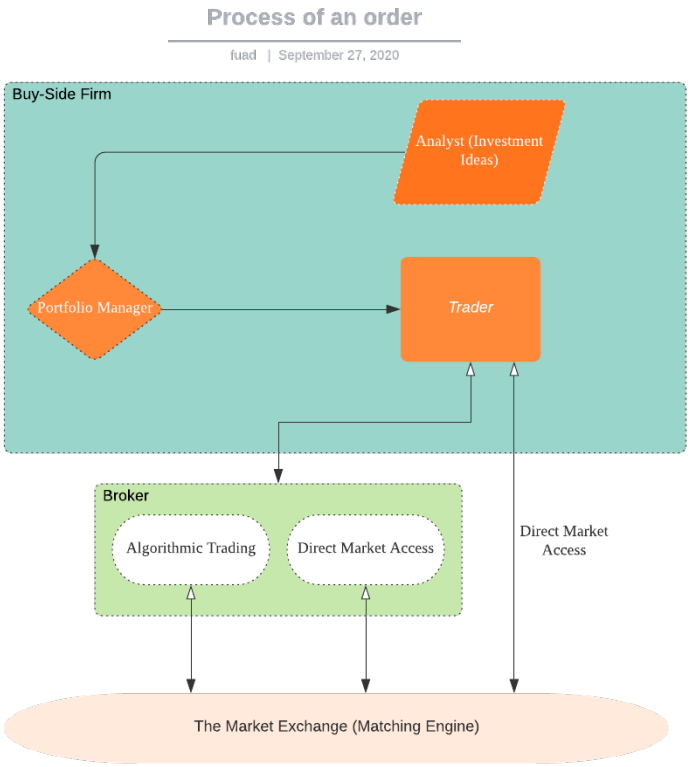


Figure 1.1 Order processing flow

After the order was paired with buyer and seller by the matching engine, it will go to the settlement system. In this system, the securities are delivered through the transaction of payment of money and the settlement date is usually 2 business days or T+2 of settlement cycle after the trade take place. Delivery and settlement of security shall be affected via electronic book entry share settlement.

Central securities depository (CSD) is where all the securities are deposited so that they are all available for clearing and settlement in one location. Through CSD, change of ownership can be easily transferred through a book entry rather than the transfer of physical certification. This was all be done through electronic which provide much faster and easier than traditional way where the physical certificate needs to be exchange whenever a trade is completed.

The Matching engine, Settlement system and CSD are all the important electronic system in market exchange infrastructure that constituted its business

process. This system will be access by another crucial part of trading process which is the electronic trading system.

1.1.2 Pre-Trade Analysis

Before any trader execute any plan to trade, they must first analyze the data first. The financial data or news was analyzed with the objective is to forecast future price movement, volatility and even generating trading signals to find trading opportunities. Common technique uses to perform this analysis are as below: -

Fundamental Analysis: This analysis focus on determining the asset's fair value or its potential future price movement. The most popular method in fundamental analysis are using ratio i.e. price to earnings ratio (P/E ratio) and Earnings Yield which is the inverse of P/E ratio. This analysis will generate trading signal once the analyst found there is different between current asset price from its fair value.

Technical Analysis: This analysis focus on predicting price movement by using historical data rather than examine underlying factors affecting asset price similar like fundamental analysis. Analyst will identify and exploit the pattern of price movement with the assumption that the price consolidates all relevant information. Most popular technique assume that price move in trends therefore they develop strategy by identifying the trend and generate trading entry signal when trend started and once the trend is expected to end they will generate exit signal.

Quantitative Analysis: This analysis focus on the price stochastic behavior which mean there is randomness and uncertainty in the price. This analysis use mathematic and statistic analysis to find suitable model to describe the stochastic behavior. One of the techniques commonly use are when quantitative use with algorithmic trading system, analyst will generate trading signals when the assets price differs with its fair value.

1.1.3 Algorithmic Trading System

Algorithmic trading (AT) is a trading system that placing order directly with trading platform without human interference. This was done based on preprogrammed trading instruction within the algorithm and trading instruction

was send within millisecond. The algorithm variable may include timing, price, quantity of the order and trigger (i.e. event) that will execute the order automatically. Below are the component of algorithmic trading

- **Algorithm**: -an algorithm is asset of rules or instruction to perform specific task in repetition order. It commonly used in algorithmic trading to find specific pattern thus generating a trading signal. Example of algorithm are sentiment analysis algorithm.
- **Automated Trading Platform (ATP)**: - An ATP is a platform where the algorithm that has been develop been executed. This platform has the ability to perform buy and sell based on the algorithm that analyst develop before. Some of the ATP has the back-testing features that will help to validate the algorithm performance before being deployed.
- **Technical Analysis**: - This analysis focus on predicting price movement by using historical data rather than examine underlying factors affecting asset price similar like fundamental analysis.
- **Back-Testing:** - Process of testing whether the algorithm is workable and verify if the trading strategy ability to deliver the result as per expectation of the analyst. Its simulate real world market based on historical data and provide analyst with understanding of the risk involved when the strategy is used in real world.

1.2 Problem Statement

Algorithmic trading in stock markets lately has generated a lot of interest among brokers and traders in the financial industry. Algorithmic trading is disruptive innovation that created a new environment where the classical way of trading deems to be slow and high risk while algorithmic trading able reduce human element and reduce risk associate with human error. In Malaysia market context, as reported by (Al-Jaifi, Al-Rassas & Al Qadasi, 2017), Malaysia has an under-develop equity market compare to their neighboring countries. The exposure to algorithmic trading is only limited to Investment Bank and Hedge fund here. Therefore, there is a wide gap between retails trader and institution traders in term of technological advantage that need to be addresses.

Retail traders in Malaysia has limited access to advance trading features, and they are exposing their investment with high market risk with trading platform that brokerage firm offer to them. Their investment often did not perform according to their expectation due to lack of knowledge, technology resources, or basic human error. This has led to dissatisfaction and causing lack of interest among retail trader to participate in equity market in Malaysia. There is a need of solution that able to implement sophisticated feature in the trading platform for retail trader to minimize their transaction cost and their exposure to market risk.

1.3 Research Questions

This project will investigate the suitability of algorithmic trading by using simple and easy to understand algorithmic trading strategy that is suitable to retail trader

- **Can algorithmic trading be more profitability than simple trading strategy such as buy and hold strategy in Malaysia stock market environment?**
- **Which algorithm perform better when generating trading signal between sentiment analysis algorithm compares with machine learning algorithm (SVM, Random Forest)?**
- **Is there any significant margin gap between sentiment analysis algorithm and machine learning algorithm performance?**

1.4 Aim & Objective

The aim of this project is to investigate the profitability of trading strategies based on algorithm trading in Malaysia stock market. Based on this motivation, the objective of this project is as followed: -

- Develop and generate trading signal using both sentiment analysis and machine learning algorithm
- Designing an algorithm trading strategy that could surpass profitability of simple strategy such as buy and hold strategy
- Test and validate the effectiveness between trading algorithm using common measure such as sharpe ratio, return over time and volatility.

Through the result of this project, it will create new insight and update the finding of previous research by including more recent data of Malaysian stock market.

1.5 Scope

This project will conduct an experiment on Malaysian stock market environment and investigating the market behavior by using trading algorithm with focus on specific algorithm such as sentiment analysis and machine learning algorithm. With this set of algorithms, the aim was to build a trigger or trading signal from pre-trade analysis and then combine with trade strategy that will act and execute buy or sell action automatically. Then comparison will be held among both of this algorithm and the performance will be observed.

1.6 Significant of Research

Due to capital market is an industry that highly regulated in this country, innovation seem to hard to come by. With the coming of disruptive technology in recent year, many firm found themselves the need to speed up their digital transformation for their organization to keep up with the competition ahead. The impact may create new risks to financial stability if not handle sooner as previous global financial crisis of 2007-2009 (Dabrowski, 2017).

This project will show the suitability of algorithmic trading in unique Malaysian stock market environment that has been discuss among local investment communities. By providing a trading strategy that utilizing algorithm such as sentiment analysis and machine learning, this will benefit both retails trader and asset manager in Malaysia as the result of this project will provide confirmation the suitability of algorithmic trading in term of profitability. Fund manager also will benefit from this project as it will provide them different perspective on their risk management strategy thus provide them with sound decision of their next investment and keep their fund profitable when compare to their competitor.

Chapter 2

Literature Review

In this project, we will used 3 trading technique where position trading will be used as performance benchmark

2.1 Position Trading (Buy/Hold Strategy)

Researcher has tested the efficiency of the buy and hold strategy such as work of (Shilling, 1992; Dechow et al.,2001; Yam, Yung & Zhou, 2009). The main reason of interest in this subject as discuss by (Fama, 1995) were that asset price reflect all available information in his efficient market hypothesis (EMH) or known as random walk thus validate the reason for use of buy and hold strategy. This has led mixed reaction among investment communities. For example (Smith & Ryoo, 2003; Borges, 2010;) and other pointed out not all market follow the EMH theory. (Loh, 2007) and (Rahman, 2019) confirm that technical analysis did not follow random walk theory as the experiment show it has significant result when comparing with buy and hold strategy.

Even though some have find that buy and hold trading strategy is not suit their style of trading, its still preferable due to key benefit such as (Sushko & Turner, 2018) state that due to its less aggressive style of investment led to low transaction cost. Therefore, there is a good practice if the investor concern about the cost associate with aggressive style of investment, (Dempster, Evstigneev & Schenk-Hoppe, 2008) conclude that buy and hold position able to protect the investor against the market volatility in the long run but at the cost of wealth opportunities.

With all the advantage of position trading, buy and hold strategy always has been set as the benchmark on any comparative and performance study by research (Shilling, 1992; Cohen & Cabiri, 2015; Hsieh, Barmish & Gubner, 2019). Its always the strategy that financial advisor will recommended to new investor due to its nature of low risk, easy to manage and accumulate growth over time.

2.2 Sentiment Analysis

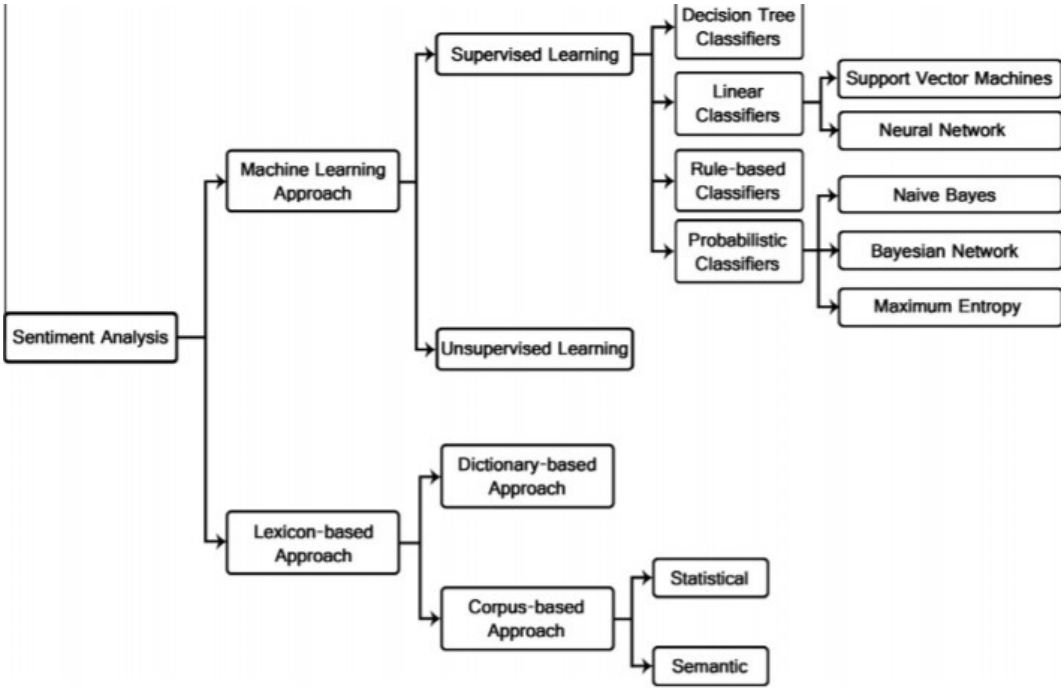


Figure 2.1 Sentiment Analysis Technique (Mehdat, Hassan & Korashy, 2014)

There are several approaches was used to understanding the behavior of stock market and price movement. The area of the research focusing on improving accuracy of prediction by conducting sentiment analysis on tweets on twitter and news report along with the stock price such as (Mehdat, Hassan & Korashy, 2014). While the research conducted by (Holden & Subrahmanyam, 2002)(Kim, Jeong & Ghani, 2014) found that there is strong correlation between financial news and prices of the stock.

All this research has limitation and drawback due to unstructured data complexity. The limitation of sentiment analysis such as sarcasm detection (Bouazizi & Ohtsuki, 2016), identifying jokes in text (Jing, Talekar & Rayz, 2018)and fake news detection(Liu & Wu, 2018).

(Uhr, Zenkert & Fathi, 2014) integrating some of text mining methods with sentiment analysis in the financial domain where he integrates with lexical resources and word association in order to analyze the news report regrading stock market. The CIWAMA concept was used to measure the association between words and simulate human word

association based on large collection of dataset and achieved good result on improving in sentiment analysis on financial news with word association.

(Shynkevich et al., 2015) uses a machine learning technique, the technique known as the multiple kernel learning(MKL), where it combine news extracted from financial news articles in order to predict of future price movement effectively. The study shows by using segmenting different categories of financial news will increase the prediction accuracy up to 79% when polynomial kernels are used on news categories. Researcher also suggests that the use of support vector machine (SVM) and K-Nearest Neighbor (K-NN) technique will have negative impact on prediction accuracy. (Poria et al., 2017), reported similar good performance on MKL but online MKL on object recognition tasks by extending online kernel learning to online MKL show that time complexity of the method dependent on the dataset itself as per research made by (Seragih, Lucy & Cohn, 2009)

(Gilbert & Hutto, 2014) had produce and develop the VADER (Valence Aware Dictionary for Sentiment Reasoning), a simple rule-based model for general sentiment analysis where it compared its effectiveness to 11 typical state-of-the-practice benchmarks, including Linguistic Inquiry and Word Count (LIWC), Affective Norms for English Words(ANEW), Senti WordNet and the General Inquirer. It is another machine learning-techniques that used the algorithms such as Support Vector Machine (SVM), Maximum Entropy, and Naive Bayes. Their study showed that VADER has improved the benefits of traditional sentiment lexicons, such as LIWC. VADER was differentiated from LIWC because it was more sensitive to sentiment expressions in social media contexts and generalized more positively when used to other domains. This was supported by (Akhtar et al., 2017). However, when use on deep sentiment analysis (Dong & DeMelo, 2018) found that VADER only lead to minor improvement.

2.3 Machine learning

Several numbers of research of machine learning approach have been study to predict the trend of stock market. Even though many machine learning techniques have been widely used (Saad, Prokhorov & Wunsch, 1996; Tome & Cavalho, 2005; Kim & Shin, 2007) none of it proved to performed better than Support Vector Machine. Support Vector Machine (SVM) has become the common technique for trader to use when refering to machine learning in algorithmic trading. SVM is a classification technique for non-linear model and has proved to shown high accuracy in many time-series forecasting. For example (Kewat et al., 2017) uses of SVM model is more accurate than classical time series forecasting and other neural network when predicting stock market. This was rejected later by (Karmiani et al., 2019) in his research, he found out that using LTSM provide much more accuracy than SVM in predicting stock movement. The finding by (Eapen, Bein & Verma, 2019)also supported that LTSM are much better at predicting stock market.

(Lee et al., 2007) has proposed a new trading framework that able to enhancing the performance of reinforcement learning based trading systems to generate buy and sell signal for investors in their day trading and maximize their profitability in the dynamic stock market. They used feedforward neural network and able to achieve 66% accuracy when use the algorithm. In (Long, Lu & Cui, 2019) find out that Recurrent Neural Network (RNN) only perform slightly better in 67% accuracy. This shown that even using most sophisticated deep learning algorithm will not guarantee a good result in predicting stock market due to the stochastic nature of the price.

Chapter 3
Research Methodology

3.1 Introduction

This chapter will be emphasized on the approach of the research on this project. There will be empirical study based on stages of data collection, data preprocessing, data exploration and data modeling & evaluation.

3.2 Research Approach

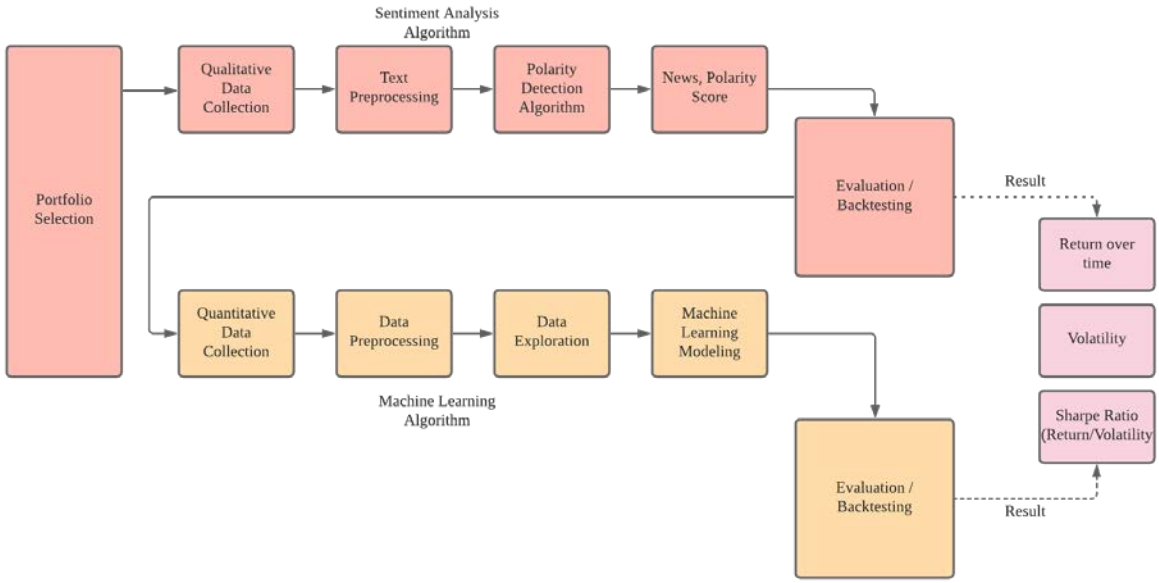


Figure 3.1 Proposed Research Design

Based on figure 3.1, these are the proposed design to run our empirical research to generate the trading signal. The research will consist of two phases. Phase one concerns the acquisition of News data (text format) which is a qualitative data then run sentiment analysis to generate trading signals. Phase two will acquired quantitative data (time series data) of daily stock price from financial website such as Yahoo Finance and run the machine learning model to generate trading signals. Then all the trading signal will go into back testing model where the strategy implementation and trading signal will be tested and evaluated.

3.2.1 Portfolio Selection

The first step in this research is to define an asset to trade. For this research we will select equity securities that traded in Bursa Malaysia Stock Exchange (KLSE). The selection of stock will be from similar industry and some stock that in different industry to add variety

3.2.2 Data Collection

3.2.2.1 Sentiment Analysis (Phase I)

For sentiment analysis, a web scraping technique will be utilized to obtain the historical news data from online news outlets. Web scraping is a process of extracting and combining computer program to obtain information on the website stored the data in systematic way. For this research we will used python library name 'Beautifulsoup' and 'Request'.

However, when using web scraping technique, cautious approach should be applied as its can cause damage to targeted server if it done aggressively. Therefore, research is required before web scraping. we would adhere the website policy and check if they allowed web scraping to their website by reviewing their website policy that website administrator put in.

3.2.2.2 Machine Learning (Phase II)

The first step in data collection part is to find the quantitative data for our model. We downloaded the data from python library by using 'yfinance' that download the data form yahoo finance website. We set the parameter that contain ticker name, start date and end date.

Ticker name is the name of the stock that we want to download for example for Top Glove Corporation Berhad the ticker name in KLSE is 'TOPG '. Then we set the start and end date of the data that we want to download which is 5 year historical data that start from 2015 until current date. Then the dataset will be stored in data frame format using another python library called Panda.

Data set that we download from yfinance will contain 7 variable which is a time series data that show the date, month and year, open which mean opening price and close mean closing price, high mean high price of the day, low mean low price, volume of transaction and the most important variable is the Adj Close which is adjusted close price that we will use to analyze historical return. Adjusted price is calculated by deducting the value of dividends from the last closing price.

Date	High	Low	Open	Close	Volume	Adj Close
2018-12-31	159.360001	156.479996	158.529999	157.740005	35003500.0	155.037109
2019-01-02	158.850006	154.229996	154.889999	157.919998	37039700.0	155.214005
2019-01-03	145.720001	142.000000	143.979996	142.190002	91312200.0	139.753540
2019-01-04	148.550003	143.800003	144.529999	148.259995	58607100.0	145.719513
2019-01-07	148.830002	145.899994	148.699997	147.929993	54777800.0	145.395203
...	...	...	...	...	...	...
2019-12-24	284.890015	282.920013	284.690002	284.269989	12119700.0	283.596924
2019-12-26	289.980011	284.700012	284.820007	289.910004	23280300.0	289.223602
2019-12-27	293.970001	288.119995	291.119995	289.799988	36566500.0	289.113831
2019-12-30	292.690002	285.220001	289.459991	291.519989	36028600.0	290.829773
2019-12-31	293.679993	289.519989	289.929993	293.649994	25201400.0	292.954712

Figure 3.2 Example of time-series dataset extracted from yfinance library.

3.2.3 Text/Data Preprocessing

3.2.3.1 Sentiment Analysis (Phase I)

The output from data collection will be in qualitative data (text format). Text format is category as unstructured data. The first step is **data transformation** where we handle it was by using tokenization technique on the document and assigning numerical number on each word in the document. Once the data was tokenized word, then we **feature selection** the data by identify the noise words that are not contributing towards classification. In this step we will remove the stop word such as white space, punctuation character, tabs, and number. In addition will we do stemming which is returning the word into it root i.e. slept into sleep, profiting into profit and such.

3.2.3.2 Machine Learning (Phase II)

Data preprocessing is one of the most important part in machine learning lifecycle. It is required to maintain the data integrity so that our data is complete, accurate and reliable when we feed it into our machine learning model.

Firstly, we will check if the data contain any missing value, if there is missing value we will applied some missing data technique such as mean imputation or multiple data imputation. Then we will smooth the noise in the data and resolve the inconsistency of the data and duplicate data.

After the data is clean, we then will perform data transformation. In the data transformation the use of data reduction, handling categorical data, dealing with imbalance data and feature engineering will be performed. Below are the technique that commonly used in data preprocessing: -

- Dealing with missing data:- Mean Imputation, Multiple Imputation, Listwise Deletion, Pairwise Deletion
- Dealing with Noise:- Binning and Regression technique
- Removing Outlier:- Clustering & Box Plot
- Data Duplication & Inconsistency:- Normalization and Indexing table
- Feature Scaling:- Mean Normalization, Z-Score
- Dealing with Categorical Data:- Label Data, One Hot Encoding
- Imbalance Dataset:- SMOTE, Under and Over Sampling
- Dimension Reduction:- Principal Component Analysis
- Feature Engineering:- Creating new feature based on domain knowledge

3.2.4 Polarity Detection Algorithm

For this polarity detection part, we will use VADER sentiment. Vader is lexical sentiment classifier and it work by labeling each of the news. Its contain a dictionary of sentiment that the word has been annotated by sentiment score ranging from -1 to 1. Its has the ability to score from sentence of word to individual word thus making it very versatile sentiment lexicon that can be used in variety of domain.

For each of the news that has been extracted and clean in data preprocessing step, VADER sentiment will organize the data in tread then classified each of the thread in

sequential order. Each tread will contain label, negative score, neutral score, positive score and compound score. The compound score is the score that use the ration of positive score and negative score in each of the tread.

3.2.5 Data Exploration

For machine learning algorithm, we will conduct exploratory data analysis (EDA) to get better understanding of our data after the cleaning process and before we feed it into our machine learning model. In the EDA, we will explore and find the hidden pattern, check assumption, test hypothesis or spot some irregularity inside the data. This will be supported by visualization of the data by using chart and graph to check of data preprocessing output performance. If there is some mistake or irregularity during the process of data preprocessing that we can spot on the visualization, the necessary step to repeat the data preprocessing step will be taken.

3.2.6 News Polarity Score

This step is the continuity from 3.2.4 Polarity Detection Algorithm subsection. Once the compound score obtain, we can now use it to visualize to find out the distribution of of the compound sentiment score. To get more clearer picture of the result we will use binning and group the score in 5 group namely Vary Negative(-1 to -0.51), Negative (-0.50 to -0.1), neutral (0), Positive (0.1 to 0.5) and very positive (0.51 to 1).

3.2.7 Machine Learning Model

For model development, because our model is using series of data point that was indexed on time, therefore the data is time series data. There are several models that we can use to perform prediction. Scikit Learn, a library for python will provide lot of model for us to test with.

Support Vector Machine (SVM) is a machine learning technique that quite popular among trader to predict future movement of stock market. SVM works by separating data point by using hyperplane as shown in figure 3.3.

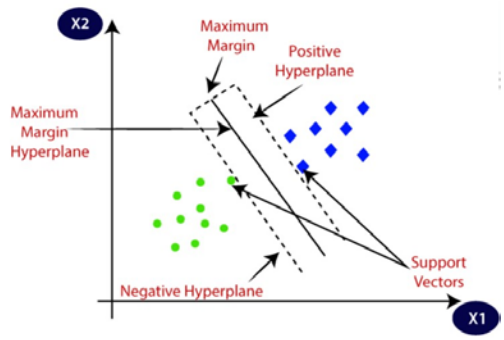


Figure 3.3 Support Vector Machine (https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm)

According to (Usmani et al.,2016), SVM when compare to other model such as RBF, Multi-Layer Perceptron and Single Layer Perceptron, archive more than 80% high accuracy compare with other model when compare side by side.

Another popular model are random forest, its another popular classification that consisting many individual decision trees and its operate similar like ensemble method. Its use wisdom of crowd similar like election process where each of nodes act as individual and give out class prediction and the class with most voted will be selected as the model prediction. (Sharma & Juneja, 2017) used random forest with LS Boost on predicting a stock market and the result was it able to outperform the more popular SVM model.

3.2.8 Evaluation and Back testing

In order to access the viability of algorithm and the trading strategy, back testing is a realistic way to perform the evaluation. Back-testing is one of the main components of algorithmic trading that simulated real world market. It used the historical data to apply the trading strategy.

For our research purpose, our result of the experiment will rely on the performance of the algorithm on the back-testing platform. This mean that both sentiment analysis algorithm and machine learning algorithm will be tested in back testing platform and evaluation will be held based on mainly on annual return and other measurement such as

- Sharpe Ratio:- calculate the ratio of return per volatility endure or in laymen's term reward/risk ratio.
- Volatility: - risk of the strategy. Higher the volatility lower the sharpe ratio.

Then the final step of the process is comparing the performance of all the algorithm trading strategy with basic strategy such buy and hold. Buy and hold is a passive strategy and one of most common trading strategies that traders use all over the world. It normally used as performance benchmark for any comparation of trading strategy (Daniel et al., 1997).

3.3 Summary

This chapter describe our approach on conducting the experiment in this project. There is 2 phases on the experiment where phase one we will get our news data (text format) and conduct sentiment analysis and generate polarity score. The polarity score will be used for either buy or sell the stock. From the polarity score, we will use it as our trading signal in the back-testing platform and get the result.

Once completed phase one, we run the second phase of the experiment, we collect our time series data from yahoo finance library then run machine learning experiment and back testing the result. When we run this experiment, we will test multiple models simultaneously and will pick the best performance of the model and select it to be test in back-testing platform. The algorithm will predict the future movement of the stock either upward or downward therefore it will set as our trading signal parameter. Once both phases completed, we will compare all the result and performance with our benchmark the 'buy and hold' trading strategy.

Chapter 4

Research Plan

Figure 4.1 Research Plan Timeline

The research plan are based on figure 4.1. where we split the whole final semester into 4 part. The capstone 2 will start with Phase I experiment by using coding the programmed in python to run the sentiment analysis. It's estimated to be completed within 5th October until 26th October 2020 timeline. Demonstration and result from this experiment will be shown to supervisor for the feedback on improvement if necessary.

Then we will move to Phase II of the experiment where will run the machine learning algorithm experiment using python code. This will be estimated take around 1 month between 2nd November until 30th November 2020 timeline. Demonstration of the working code and result will be shown to supervisor for feedback on improvement if required.

Once all the experiment has successfully been conducted, documentation process will started on 7th December until 28th December 2020. On this part, all the result will be documented in proper format and proofread and all the necessary paperwork including form such as declaration of thesis confidentiality, declaration of supervisor, declaration of originality and exclusiveness are signed before submission.

And the final part is submission of the document will be done within 4th January until 29th January 2021. Within this period, we will start to prepare the presentation content and engage with the preparation for the viva voce.

Chapter 5

Methodology

5.1 Introduction

This chapter will start with additional topic and correction on previous topics discuss on Chapter 1 then it focuses on every stage of our model implementation together with the detailed explanation of the method and technique used in implementing the experiment. Google Colab was chosen as our experiment environment. This experiment used two type of dataset and the justification of the choosen company stock will be explained in detail. Not only that, but variety of technique also used in this experiment including data processing, data exploration, feature selection and together with their step by step is included and clarify. These processes are important before we feed our data into the model algorithm to increase the performance therefore the shape of the data or the input feature must be set before the algorithm can analyze it.

5.1.1 Research Gap

Many researchers have done their trading strategy based on foreign market such as DJIA, Hang Seng, London Stock Exchange. Not many researchers have done research on Malaysia market particularly with the used of algorithmic trading. While local trader mostly using conventional technique such as buy and hold strategy, this paper will explore various algorithmic trading strategy with the highlight of use in particular event-based sentiment analysis.

Previous research about sentiment analysis used in stock market but most study focus on using the sentiment score directly to generate the trading signal. This study will experiment with hybrid used of sentiment analysis and focus on combining sentiment analysis score signal with moving average from Simple Moving Average statistical model.

5.1.2 Research Question

This project will investigate the suitability of algorithmic trading by using simple and easy to understand algorithmic trading strategy that is suitable to retail trader.

- **What are the algorithmic trading strategy option available for retails trader in Malaysia?**
- **Which algorithm perform better than simple Buy and hold Strategy?**

- **How to implement sentiment analysis in algorithmic trading effectively?**

5.1.3 Research Objective

The aim of this project is to investigate the profitability of trading strategies based on algorithm trading in Malaysia stock market. Based on this motivation, the objective of this project is as followed: -

- To explore several candidate algorithm strategies that has potential to performed better yet simple to execute by retail trader.
- To identify high performance algorithmic trading strategy that is better than conventional buy & hold strategy.
- To recommend effective way to implement SA in algorithmic trading.

Through the result of this project, it will create new insight and update the finding of previous research by including more recent data of Malaysian stock market.

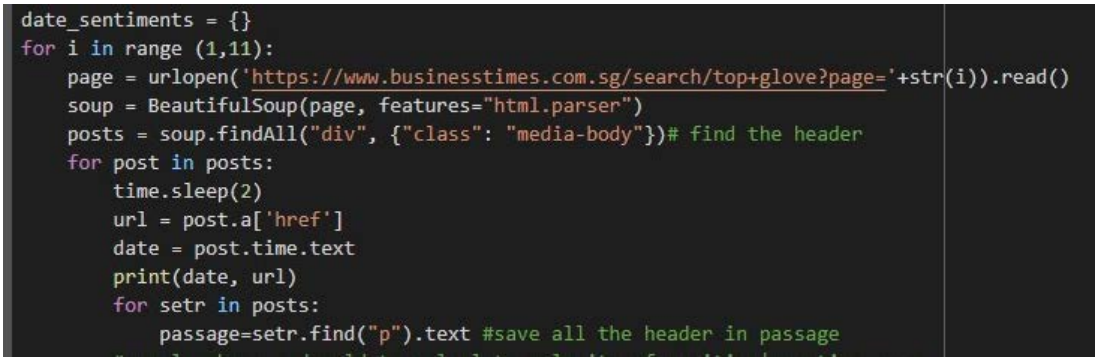
5.2 Data Collection

5.2.1 Sentiment Analysis

For sentiment analysis, the data was collected from the business news portal, BusinessTimes.com by using web scraping technique. BusinessTimes news portal cover business and economic news on South East Asia and it based in Singapore. This website was chosen as this project main news portal due to its very friendly interface and easy to extract data using web scraping compare to other news website that we find its difficult to obtain data due their internal policy that will be covered in next section.

Web scraping is a data scraping technique in which the computer program extracts the data from html website. Its use similar technique of web crawler automatic indexing process that was long used by Google or Yahoo Search. However, instead of indexing the website, web scraping collect data from the website. Web scraping is not the only way to extract data from the website. The other way was by using Application Programming Interface (API) data that was provided by the website itself. However, the only drawback of using API is the data that provided to the public is limited, expensive or non-existence (no API provided at all). There is certain consideration when extracting data from website that web scraper need to aware of such as permission to obtain the data in their website policy. There is some website that forbid their data to be extract due to the data protection policy and the use of bandwidth required when web-scraping the data that can cause the server to overload if it's doing wrongly or abusive in nature.

For us to do web scraping, there are 2 components required to do the web scraping. They are as **Figure 5.2.1** below:



```
date_sentiments = {}
for i in range (1,11):
    page = urlopen('https://www.businesstimes.com.sg/search/top+glove?page='+str(i)).read()
    soup = BeautifulSoup(page, features="html.parser")
    posts = soup.findAll("div", {"class": "media-body"})# find the header
    for post in posts:
        time.sleep(2)
        url = post.a['href']
        date = post.time.text
        print(date, url)
        for setr in posts:
            passage=setr.find("p").text #save all the header in passage
```

Figure 5.2.1 Web Scraping Code Snippet

1. Urllib library- its contain module such as urllib.request that contain function urlopen() that is required for us to open URL within our program. Another module that we use is urllib.error that will handle any common error when interacting with URL such as Error 403 which will result our program to break. The output of this component is text format of the HTML.
2. BeautifulSoup library- This library is to parse HTML pages. Even though we can use other method to parse the HTML pages using find() function or regular expression but BeautifulSoup library will make parsing html a lot more easier. The way it worked is by search specific tag in the html document by providing the argument in find_all() function and then extract the data from the specific tag

5.2.2 Machine Learning

For machine learning technique, the data was collected from Yahoo Finance by using opensource API. The yfinance library in python will collect automatically all historical market data of the particular stock. The library, yfinance was created by Ran Aroussi and the API is distributed under the Apache Software License. Yahoo finance is the only free source of historical market data that we can use in Malaysia as the alternative data required subscription fees.

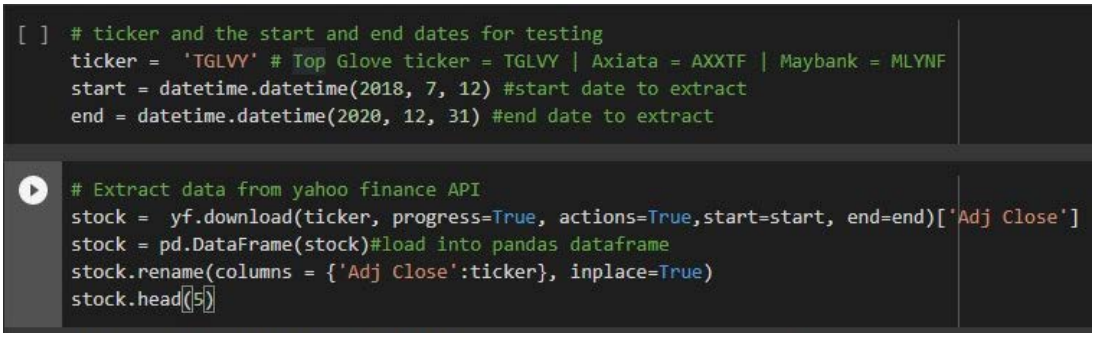


Figure 5.2.2 Data collection code

As per **Figure 5.2.2**, extracting data from API is much easier compare to other method such as web scraping as the data already provided by the website for public consumption. The only component in the API that we need to specified is:

1. Ticker - 2 to 5 letter that represent the securities listed in the exchange. For example, in **Figure 5.2.2**, show that the ticker TGLVY represent Top Glove security in KLSE.

2. Start - We need to specify the exact date when we want our time series data to start.
3. End - To stop extracting the time series data at certain point of date.

For this experiment, we choose Top Glove (Ticker: TGLVY) stock as our sample stock and build algorithmic trading strategy. There are several reasons that we choose this particular stock such as it's the most traded counter in 2020 due to increase demand of glove around the world. Another reason was that this company stock is one of the companies that face what is known as Black Swan Event. Black Swan event is an unpredictable event beyond what is normally expected and rare by nature but still highly likely to happen such as Covid-19 outbreak.

Due to this reason, building algorithmic trading strategy for Top Glove possesses some serious challenge and its required testing various trading strategy that robust enough to deal with the Black Swan event. Black Swan investing is by nature is a bearish trading that associate with falling share price and this is the opportunities for us to understand more of stock behaviors when facing unpredictable event in general.

5.3 Data Preprocessing

5.3.1 Dataset Explanation

```
print(page)

b'<!DOCTYPE html>\n<!--[if lte IE 8]><html class="no-js lt-ie9">
p://ogp.me/ns# article: http://ogp.me/ns/article# book: http://og
roduct: http://ogp.me/ns/product# content: http://purl.org/rss/1.
rdfs: http://www.w3.org/2000/01/rdf-schema# sioc: http://rdfs.org
s/core# xsd: http://www.w3.org/2001/XMLSchema# schema: http://sch
b">\n  <meta charset="utf-8">\n  <meta http-equiv="X-UA-Compatibl
```

Figure 5.3.1 Web Scraping Data before preprocessing

For Sentiment analysis, the data we received during web scraping was in Hypertext Markup Language (HTML) format as per **Figure 5.3.1**. Its required pre-processing to be useful for our system. BeautifulSoup library will assist in parsing the HTML document as previously explain in data collection subsection.

Date ▲	Open	High	Low	Close	Adj Close	Volume	Divide…	Stock …
2016-06-24T00:00:00.000	2.2750000954	2.2750000954	2.2750000954	2.2750000954	2.049366951	0	0	0
2016-06-27T00:00:00.000	2.2750000954	2.2750000954	2.2750000954	2.2750000954	2.049366951	0	0	0
2016-06-28T00:00:00.000	2.2750000954	2.2750000954	2.2750000954	2.2750000954	2.076290369	0	0.0295	0
2016-06-29T00:00:00.000	2.2750000954	2.2750000954	2.2750000954	2.2750000954	2.076290369	0	0	0
2016-06-30T00:00:00.000	2.2750000954	2.2750000954	2.2750000954	2.2750000954	2.076290369	0	0	0
2016-07-01T00:00:00.000	2.2750000954	2.2750000954	2.2750000954	2.2750000954	2.076290369	0	0	0
2016-07-05T00:00:00.000	2.4500000477	2.4500000477	2.4500000477	2.4500000477	2.2360050678	2000	0	0
2016-07-06T00:00:00.000	2.3250000477	2.4249999523	2.3250000477	2.4249999523	2.2131881714	2000	0	0
2016-07-07T00:00:00.000	2.4249999523	2.4249999523	2.4249999523	2.4249999523	2.2131881714	0	0	0

Figure 5.3.2 Top Glove dataset from Yahoo Finance

For Machine learning phase, the market data was acquired from yahoo finance library. As per **Figure 5.3.2** the explanation of each column is as followed:

- **Date: -** The date of the opening day of the particular stock. Stock exchange only open on working day so there is 252 day annually.
- **Open: -** Opening price of the trading session of day. The market open at 9.00 am
- **High: -** The peak/highest price of the stock movement on the day.
- **Low: -** The lowest price of the stock movement on the day
- **Close: -** The closing price of the day. Market close at 5.00 pm
- **Adj Close: -** The adjusted closed after dividends, stock split that sometime happen with securities that pay dividend or experiencing stock split event. ADJ Close give more accurate closing price rather than Close.
- **Volume: -** Volume is the total share that change ownership for the particular day

- **Dividends: -** Dividend is cash paid to investor per holding of the securities
- **Stock Split:-** When the company issues more shares of stock thus lower the value of each shares.

5.3.2 Preprocessing

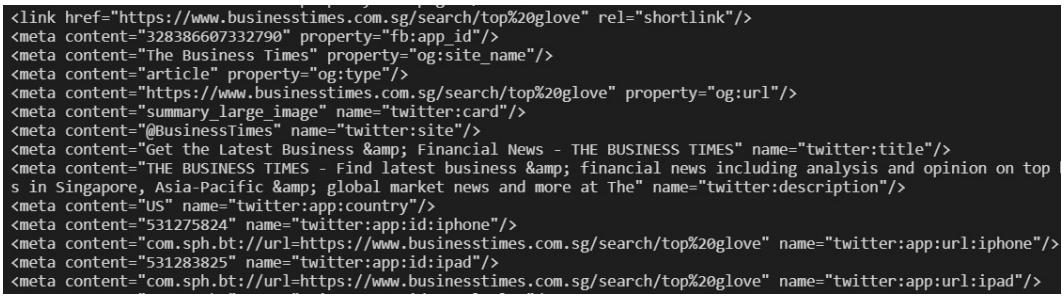


Figure 5.3.3 The HTML Tags & Metadata

During web scraping preprocess, BeautifulSoup function will return a string stripped of any HTML tags and metadata as per **Figure 5.3.3**. From there, we then find the specific paragraph in the HTML tag that contain headline paragraph that will be used as input for our sentiment analysis. Once the headline is found, the next step is to save it into text. Tokenization of the text was required before feeding it to sentiment analysis from pysentiment2 library. This library was based on work of (Loughran & McDonald, 2016) that specialize in accounting and finance lexicon. This sentiment analyzer will compound the score of our news headline into range of -1 for negative sentiment, 0 for neutral and 1 for positive sentiment. Final input of sentiment analysis will be discuss in data exploration later.

Scale	No. of words	Sample words
Negative	2,337	termination, discontinued, penalties, misconduct, serious, noncompliance, deterioration, felony
Positive	353	achieve, attain, efficient, improve, profitable
Uncertainty	285	approximate, contingency, depend, fluctuate, indefinite, uncertain, variability
Litigiousness	731	claimant, deposition, interlocutory, testimony, tort
Weak Modal Words	27	could, depending, might, possibly
Strong Modal Words	19	always, highest, must, will

Figure 5.3.3.1 Loughran and McDonald Financial Sentiment Dictionary (source: https://www.kovcomp.co.uk/wordstat/Sentiment.html)

- Loughran & McDonald (LM) Financial Lexicon: - The use of LM instead of Vader lexicon stated earlier was because we found that the Vader lexicon is more suitable to general lexicon instead of financial lexicon. There are several words that tag negative in general lexicon but typically not negative in financial context. This

will impact our accuracy in generating the signals. **Figure 5.3.3.1** are the sample word from LM dictionary and the scale classification.

For market data that we extracted from Yahoo Finance, its only required minimal preprocessing as the data is already clean and the only preprocessing that was done just changing the column name from Adj Close into the ticker name (TGLVY).

5.3.3 Feature Selection

```
# Extract data from yahoo finance API
stock =  yf.download(ticker, progress=True, actions=True,start=start, end=end)['Adj Close']
stock = pd.DataFrame(stock)#load into pandas dataframe
stock.rename(columns = {'Adj Close':ticker}, inplace=True)
stock.head(5)
```

```
[**********100%**********]  1 of 1 completed
```

Date	TGLVY
2018-07-12	4.329658
2018-07-13	4.329658
2018-07-16	4.329658
2018-07-17	4.817339
2018-07-18	4.817339

Figure 5.3.4 Data extraction and feature selection

During market data extraction from Yahoo Finance, we only extract Date column and Adjusted Close column only as per **Figure 5.3.4**. This variable was selected as we want to created univariate data. The reason of this because the objective was to find the price movement and this feature (ADJ Close) are the only feature provide us with last price of stock in the day. After that we save our data in Pandas dataframe for easy manipulation of the data later.

5.4 Data Transformation

Due to there is no input (x) and output(y) feature in univariate data that required for supervised learning, we need to build an input(x) using feature engineering to predict the movement of stock. Feature engineering will convert time series into supervised learning problem that can be used on supervise learning algorithm such as Support Vector Machine, Decision Tree, Random Forest or Multi-Layer Perceptron model.

5.4.1 Feature Engineering

```
[ ]  # calculate daily log returns and market direction
     stock['returns'] = np.log(stock / stock.shift(1))
     stock.dropna(inplace=True)
     stock['direction'] = np.sign(stock['returns']).astype(int)
     stock.head(5)
```

Date	TGLVY	returns	direction
2018-07-13	4.329658	0.000000	0
2018-07-16	4.329658	0.000000	0
2018-07-17	4.817339	0.106733	1
2018-07-18	4.817339	0.000000	0
2018-07-19	4.817339	0.000000	0

Figure 5.4.1 Creating binary direction from data

To perform feature variable to predict the market direction, we create a direction column by calculate the daily log return (log [current price/original price]). Then we drop any element that contain missing value and keep the dataframe with valid entries in the same variable. And finally we find the market direction by using numpy sign () function which will return value (<0=-1 , 0=0, >0=1). The code and result as per **Figure 5.4.1**

2018-07-26	0.000000	0.000000	0.000000	0.000000	0.000000	1	1	1	1	1
2018-07-27	-0.001492	0.000000	0.000000	0.000000	0.000000	0	1	1	1	1
2018-07-30	-0.077070	-0.001492	0.000000	0.000000	0.000000	0	0	1	1	1
2018-07-31	0.000000	-0.077070	-0.001492	0.000000	0.000000	1	0	0	1	1
2018-08-01	0.000000	0.000000	-0.077070	-0.001492	0.000000	1	1	0	0	1

Figure 5.4.2 Binary Direction Lag Return

We then define the lag return that we want to use. The reason of lag is used because it ability to turn time series into supervised learning problem. Lagged return train the model to use previous return pattern as the input variables. For this experiment, we

will use 5 lags as our input. 5 lag was choose as it’s commonly use as lag for short term trading. Shift() function will help to create this lag return for our univariate data. Once the lag was calculated, we converted it into binary and the final output of lag are shown as in **Figure 5.4.2**.

5.5 Data Exploration

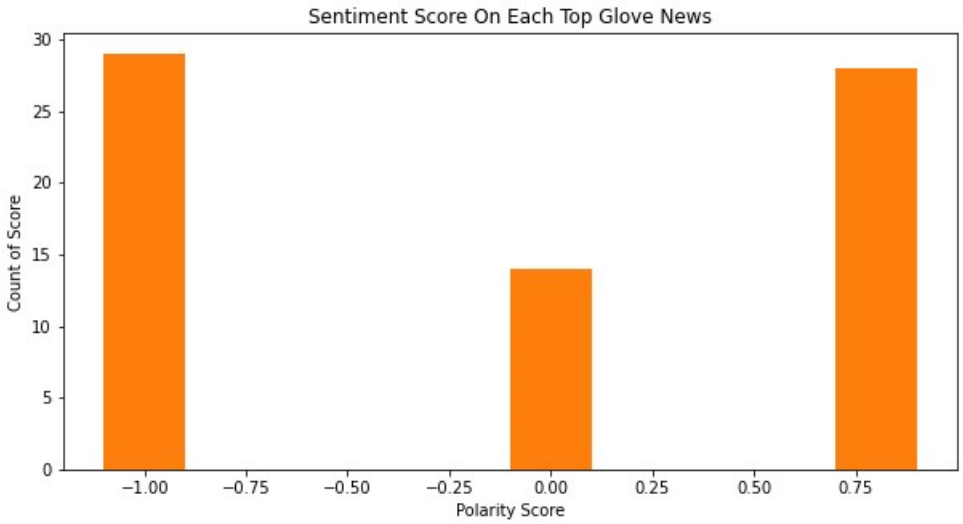


Figure 5.5.1 Histogram of Sentiment Analysis Score

From the result of our sentiment analysis, its showed that our sentiment analysis have 3 output as illustrated in**Figure 5.5.1.** There are negative sentiment, neutral sentiment and positive sentiment. The news headline sentiment was extract during period of 12 July 2018 until 18 January 2021 from Business Times Singapore website. The output from sentiment analysis will be used later as trading signal together with SMA. This signal that will indicated whether to buy the stock and established open position if the sentiment is positive or to sell the stock and closed the position in our trading strategy. If the sentiment is neutral, its signal hold position. The count of negative and positive seem to be the same (29 on negative score, 28 on positive score and 14 on neutral score).

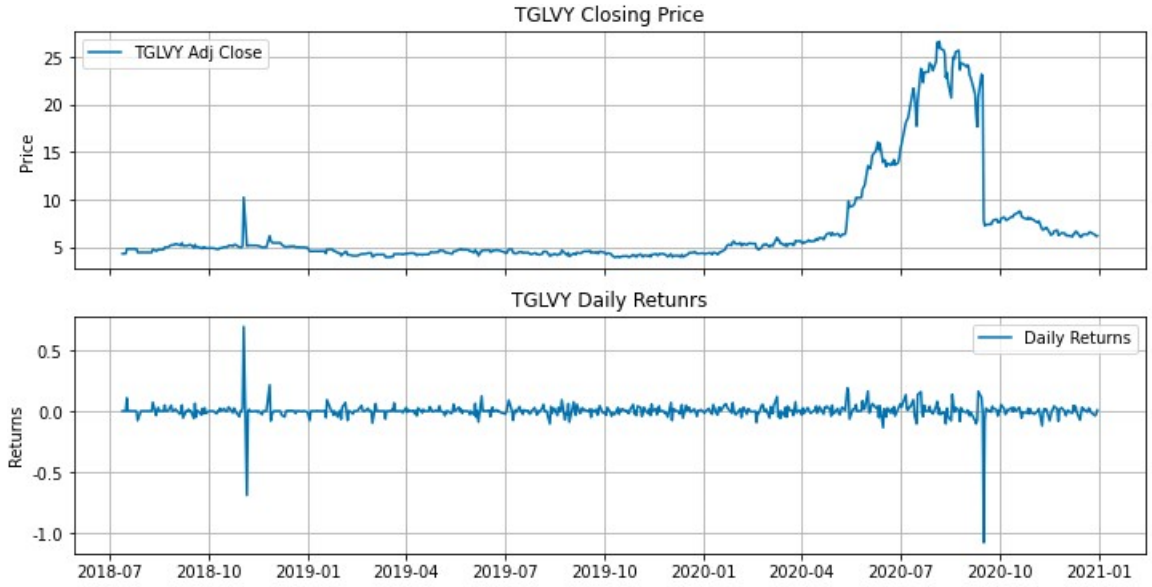


Figure 5.5.2 Top Glove Closing Price & Daily Return

Figure 5.5.2 showed Top Glove stock movement from July 2018 until January 2021. On the top picture show the price movement of the stock while on the bottom is the calculated daily return of the stock that we have engineered during data transformation step using closing price input. From the line chart, the average pattern of price is within range of RM 5 while return normally are within the range of 0.5 and -0.5 except for 2 occasion. This show that the price and returns are correlated. When analyze furthermore, the first surge of price was on 2018 was due to their report earning that has positive look and the second surge was due to covid-19 demand of medical equipment. As mention before, value of returns and price has some correlation between them and to amplify these signals, we will use these variable to engineered new variable (lag return) that will act as our input (x) & output (y) in supervised learning model for our univariate time series data.

.

5.6 Model Development

The task to find the best strategy for algorithmic trading is much depend on the choosen model. The goal of the model is to find and generate what is commonly known as alpha. Alpha is defined as excess portfolio returns of the benchmark used for evaluation. This can be translated as the model must be able to generate more profit than the benchmark. In this experiment, we will use simple strategy such Buy and Hold Strategy as the benchmark for all the model to beat.

For Phase I experiment, sentiment analysis score that has been generate before will be used together with SMA statistical model. For phase II machine learning, Logistic Regression, Gaussian Naive Bayes, Support Vector Machine (SVM), Random Forest, and Multi-Layer Perceptron (MLP) algorithm is choosen to predict the market direction based on the lag input that we had created on feature engineering to predict the market direction y output (-1 or +1).

5.6.1 Simple Moving Average (SMA) + Sentiment Analysis

For Sentiment analysis phase I experiment, we will construct SMA model to smooth the data and able to determine its support and resistance levels based on moving average period. Moving average is calculated by adding the price over a number of time periods and then dividing the sum by n time periods as shown in **Figure 5.6.1**

$$SMA = \frac{A_1 + A_2 + \ldots + A_n}{n}$$

where:

$A =$ Average in period n

$n =$ Number of time periods

Figure 5.6.1 Simple Moving Average Formula

Based on the rules, whenever there is positive sentiment score and the price is above moving average line it will generate buy signal while if the stock was moving inversely it will generate sell signal. However, we need to find moving average period that maximize the return. We will run hyperparameter search with an algorithm to able find the best moving average period for us. Below is the pseudocode of our algorithm:

Step 1: - Define daily data of our stock by selecting the timeframe.

Step 2: - Split the dataset to training & testing set.

Step 3: - Run hyperparameter search to find the optimal period by applying different moving averages period on the training set and each one will calculate the average return value after N days when the close price is over the moving average line.

Step 4: - then choose moving average length that maximizes such average return.

Step 5: - Repeat the step for test Set

Step 6: - Validate the result by using MSE and RMSE

Once we find the period that will provide maximum return for this model, we then incorporate it with the score from sentiment analysis. Then visualization will be used to understand the output.

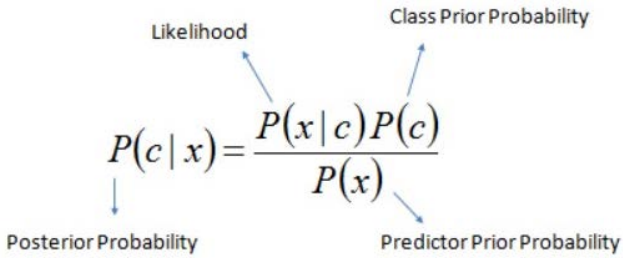
5.6.2 Linear Regression

Linear regression is a technique in statistics that use linear approach to model relationship between the dependent and independent variable and is represented in equation of Y=a+bX. Y is the dependent variable and also a constant value while X is the independent variable and b is the slope of the regression line.

Linear regression normally used for regression problem however in this experiment it will be converted to be used as our classification problem. Its work in simple way for example, if the output is 0.7, then there is a 70% chance that tomorrow's closing price is higher than today's closing price and the model classify it as 1. Else if the output is negative value, it will classify as -1 which mean it will generate sell signal.

5.6.3 Gaussian Naïve Bayes

Another popular technique to build algorithmic trading strategy is the Gaussian Naïve Bayes. Its popular because of its low computation cost and memory requirements to facilitate large and high dimensional dataset. Naïve bayes model involves a simplifying condition independent assumption and in our model the price direction class (negative and positive) show that its conditionally independent from each other. The algorithm is as per **Figure 5.6.2**.



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Figure 5.6.2 Gaussian Naïve Bayes Algorithm

P(c|x) is the posterior probability of target given the attributes, *P(c)* is the prior probability of class, *P(x|c)* is the likelihood of predictor given class and finally the *P(x)* is the prior probability of predictor.

5.6.4 Support Vector Machine

Support vector machine (SVM) is another machine learning algorithm that is popular due to its able to produce significant accuracy without using too much computation power.

SVM algorithm work by finding the best line or decision boundary and this boundary is called hyperplane. Support vector is the closest point of the line from both classes and between the line between vector and hyperplane is the margin. SVM algorithm goal is to find the maximum margin between this point and its called optimal hyperplane.

The different between SVM and other machine learning algorithm is SVM do not focus on minimizing errors unlike other algorithm but its focus on structural model that has least risk of making mistake on future data.

5.6.5 Random Forest

Random forest is an ensemble learning method that derived from the decision tree model. Its usually train with bagging method which is a combination of learning models to increase the overall result. The important hyperparameter in random forest to increase the performance is n_estimator and max_depth. As per documentation of Sklearn, the n_estimator is the number of trees in the forest. In general, the higher number of trees the higher the performance and stable the model but at the expense of computation time. For the machine learning phase II experiment, we will find the optimal hyperparameter manually by tweaking the value of n_estimator & max depth.

5.6.6 Multi-Layer Perceptron

Multi-layer perceptron (MLP) is an artificial neural network that is used feedforward network. MLP contain 3 layers of nodes consist of an input layer, hidden layer and output layer.

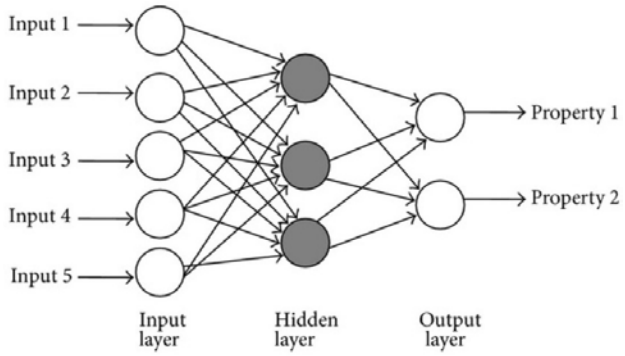


Figure 5.6.3 Multi-Layer Perceptron with single hidden layer (source: Djuriš et al. 2012)

As per **Figure 5.6.3,** input layer receives the signal from the source, then in between, single or multiple hidden layers act as computational engine for the MLP and finally an output layer that makes a decision or prediction about the input. Training the MLP involves adjusting the hyperparameters or the weights and biases depending on the model to minimize error and increase the accuracy. For the machine learning phase II experiment, we will find the optimal hyperparameter manually by tweaking the value of activation function, solver, max_iter and the details will be explain in the result.

The hyperparameter that we going to experiment is consist of number of neurons in hidden layer, activation function, regularization and learning rate. The simple explanation is as followed: -

- Hidden Layer: - For hidden layer, we use single hidden layer to avoid complexity but we will experiment with number of neuron within the hidden layer. The default neuron is 100 so we will try decrease and increase the default and see which performed better
- Activation Function: - The aim to find best activation function and by default the activation is Relu and Adam. Other activation function including Tanh, SGD, RMSprop and Adagrad that will be experiment to find best optimize function.

- Regularization: - Regularization is to avoid overfitting on our model. Sklearn provide L2 regularization which also known as ridge regression. The default value is 0.0001 and we will experiment with multiple value to find best output.
- Learning Rate: - The default is 0.001 and we will experiment with different value such as 0.01 or 0.1.

5.7 Evaluation & Validation

Each time the model being developed, accuracy and other performance evaluation metrics will be evaluated. For experiment phase 1, since SMA is forecasting a regression problem, Mean Square Error (MSE) and Root Mean Square Error(RMSE) will be use for performance measurement. We will utilize scikit learn function mean_square_error() function and then use math function of sqrt() function and use the output of MSE as input in our function to calculate the RMSE value. For our machine learning phase II, we will use accuracy score, precision, recall and f1-score to calculate all 5 machine learning algorithm performance.

After the performance evaluation, the model will be validated to confirm that the output of the model performance are at acceptable level when using real data. Backtesting simulates our strategy based on historical data with objective to achieve performance result that generalize to new market condition. Backtrader is the python library that is popular for backtesting and even some trading house use Backtrader to prototype and test their new strategy before deploying them into their platform.

The performance matric that will be used in our backtesting are as followed.

- **Annual Return** -return on year basis
- **Cumulative returns**- total aggregate return within the observed time frame
- **Volatility** - statistical measure of the dispersion of returns for a stock, higher the value higher the risk.
- **Max Drawdown** – highest percentage loss from previous peak.
- **Sharpe Ratio** – measure of risk adjusted return. High Sharpe ratio are often desirable.
- **Rolling Sharpe Ratio** – Similar to Sharpe Ratio but in 6-month windows instead of overall time frame. Useful to pinpoint event on when the strategy is outperform/underperform within the observed periods.
- **Daily Value at Risk** – Inform investor the risk they taking in daily basis.

Most of the investor only focus on Sharpe Ratio to measure the performance of their strategy. A Sharpe Ratio of 1 means that the investment has equal return and the risk being taken and if its less than 1 means that more risk is being taken compare with reward in return.

Chapter 6

Result and Analysis

6.1 Introduction

In this chapter, the finding and performance of all the model result will be presented. We will test 6 candidate strategy namely SMA + Sentiment Analysis, Logistic Regression, Gaussian Naïve Bayes, Support Vector Machine, Random Forest and Multi-Layer Perceptron. All of the models will be evaluated and backtest to simulate real time environment thus will critically analyzed. The causal relationship behind each candidate will be discuss in detail together with individual evaluation and comparative analysis. Finally, the performance of the candidate strategy will be compared with previous work of researcher in the same field.

6.2 Baseline Benchmark

Start date	2018-07-12
End date	2020-12-30
Total months	29
	Backtest
Annual return	15.6%
Cumulative returns	43.2%
Annual volatility	103.1%
Sharpe ratio	0.68
Calmar ratio	0.20
Stability	0.35
Max drawdown	-77.3%
Omega ratio	1.23
Sortino ratio	1.10
Skew	3.60
Kurtosis	110.20
Tail ratio	1.09
Daily value at risk	-12.7%

Figure 6.1 Baseline Model Buy and Hold Model

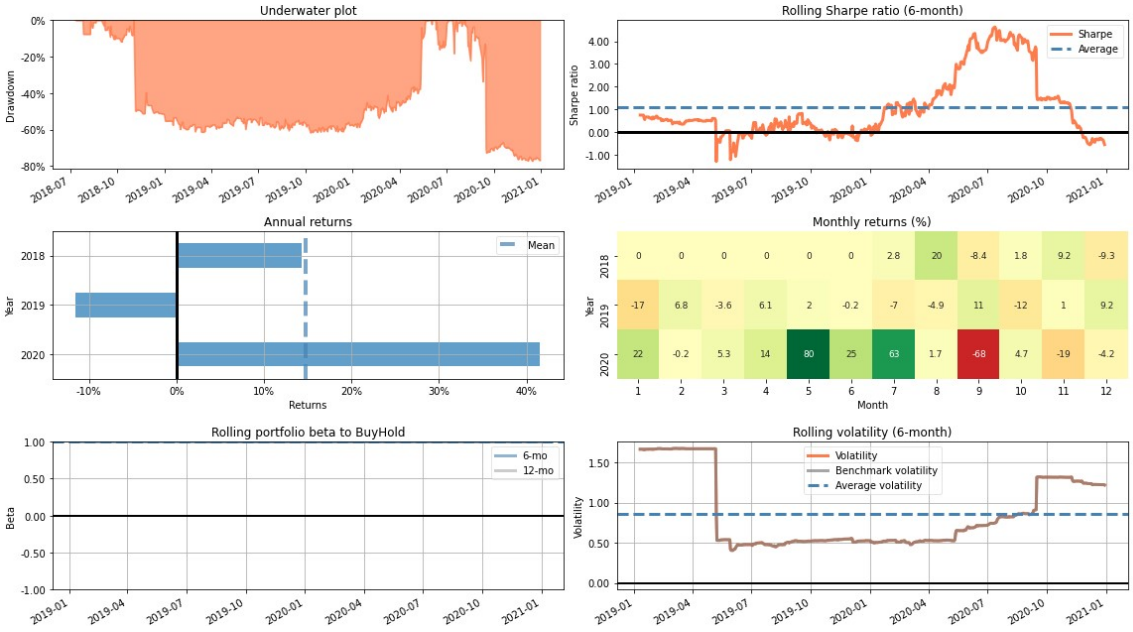


Figure 6.1.1 Baseline Performance Chart

For our baseline benchmark, Buy and Hold strategy is coosen. Buy and hold strategy is the simplest trading strategy an investor can used as the concept is buy and forget. Most academic (Cohen & Cabiri, 2015; Hsieh, Barmish & Gubner, 2019; Abbasi, Samavi & Koosha, 2020) and active investor used this strategy as their benchmark. The objective for our backtesting is to find model that can outperform our key metrics of baseline benchmark. The metrics to beat by all 6 models are the return (annual return & cumulative return), volatility (in form of annual volatility & maximum drawdown), and the risk adjusted return (in form of Sharpe Ratio, rolling Sharpe ratio & daily value at risk).

From the **Figure 6.1**, the baseline returns for this experiment is 15.6% in annual return and 43.2% in cumulative return (both higher the better). The annual volatility is 103.1% and maximum drawdown 77.3% which lower the value of the risk, the better of model performed. The risk adjusted Sharpe Ratio is 0.68, average rolling Sharpe ratio at 1.00 (higher the better) and daily value at risk is 12.7% (lower the better).

6.3 Phase I Result

Phase I is our experiment using sentiment analysis score as our trading signal. We then combine the signal with SMA model to enhance the model performance.

6.3.1 Hyperparameter Search & Training the Model

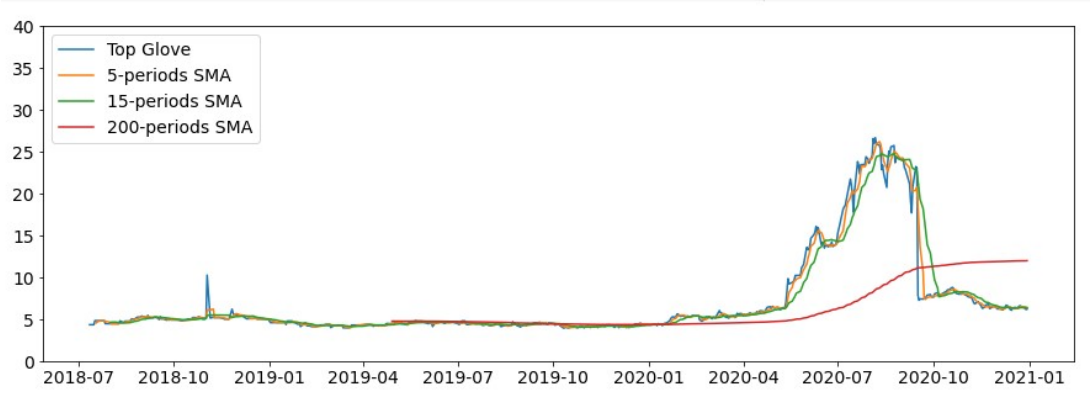


Figure 6.2 Moving Average of 5, 15 and 200

N_Forward	Test Return	Train_Return	RMSE	MA_Period
5	0.05	0.01	3.05	27
10	0.11	0.03	4.51	25
15	0.17	0.05	5.80	25

Table 6.1 Evaluation result for SMA Model

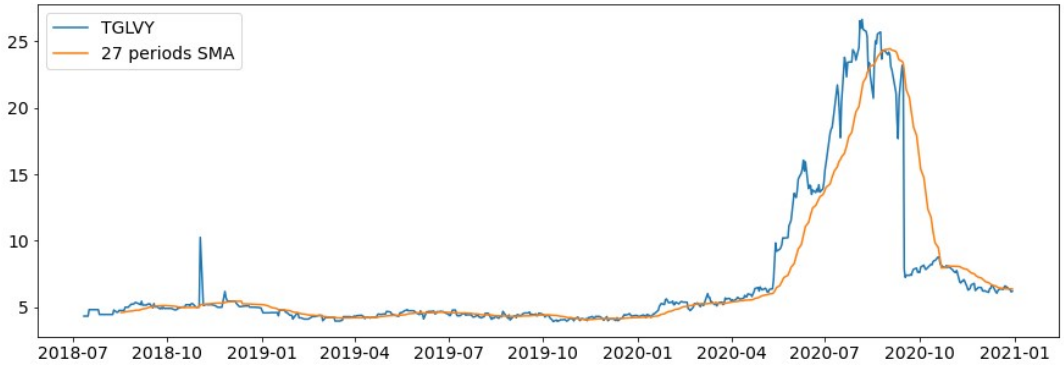


Figure 6.2.1 Moving Average after hyperparameter search

In **Figure 6.2** illustrated the commonly used moving average period in the industry. 5 moving period is commonly used by day trader while 15 moving average period is commonly used by swing trader and finally longer moving average of 200 normally been utilize by long term trader. To find the best moving average period for our model, we conduct hyperparameter search to find it by implement for loop that range

from 5 moving average until 500 moving period. The hyperparameter search found that 5 n_forward (open position) is the best for our model with RMSE is low compare to other. Its also indicate that we need 27 moving average period to obtain lowest RMSE for our model as per **Table 6.1.** And from the visualization of **Figure 6.2.1** our moving average line is closed to closing price and there is a lot of crossover point (when price cross the moving average line) that we can used as buy signal.

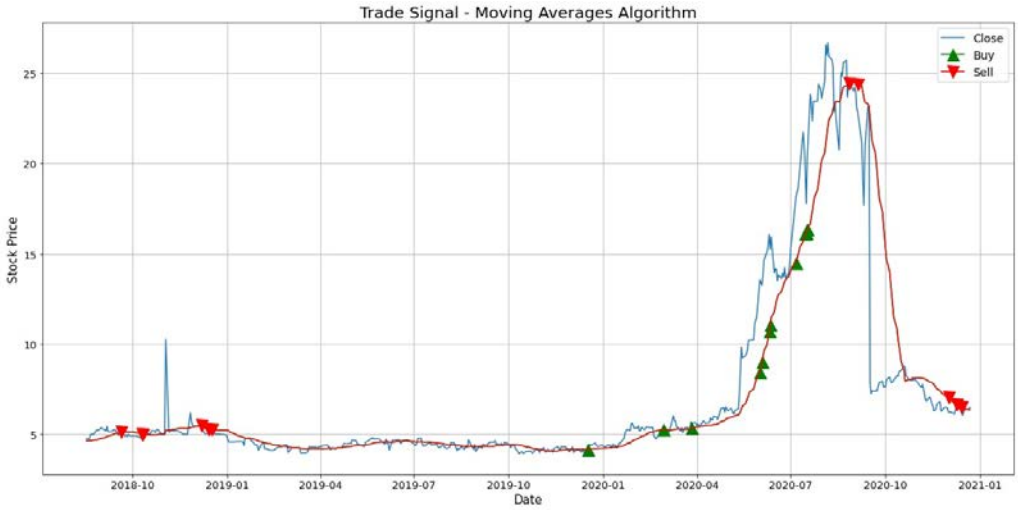


Figure 6.2.2 Trading Signal of SMA + Sentiment Analysis

Figure 6.2.2 illustrated the combination of sentiment analysis score and moving average line that we build to generate signal. When combined, we able to reduce the crossover signal which in other word mean that the model will not always generate signal whenever the price crosses the moving average line. Instead, it required additional information before it can generate the buy/sell signal. Based on the rules we set earlier, the positive/negative score from sentiment analysis will be the additional rules it needs to comply before generating the signal. Reduce crossover signal will save trader the transaction cost paid to brokerage house. There is now only 9 sell signal and 9 buy signal.

6.3.2 Backtesting

Start date	2018-07-12
End date	2020-12-30
Total months	29
	Backtest
Annual return	26.2%
Cumulative returns	77.9%
Annual volatility	59.6%
Sharpe ratio	0.80
Calmar ratio	0.37
Stability	0.48
Max drawdown	-71.2%
Omega ratio	1.39
Sortino ratio	1.03
Skew	-7.21
Kurtosis	142.01
Tail ratio	1.40
Daily value at risk	-7.3%

Figure 6.3 Tear Sheet Summary Result

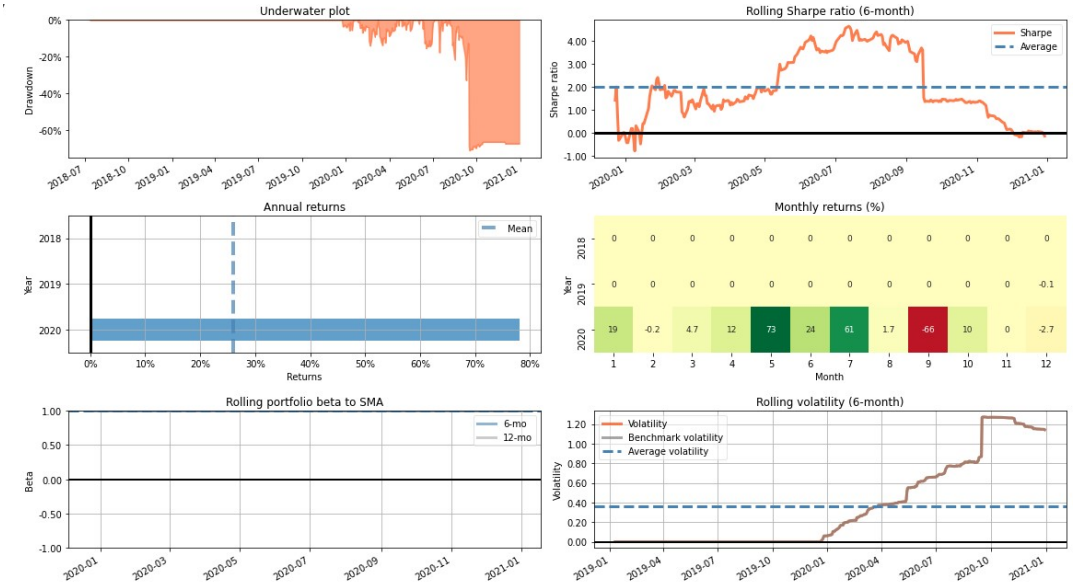


Figure 6.3.1 Backtesting Performance Chart

Figure 6.3 and **Figure 6.3.1** is backtest result of our model. The annual return of this model is 26.2% and cumulative return 77.9%. For risk adjusted return in Sharpe Ratio the score is 0.80%. But when we see the 6-month rolling Sharpe Ratio on **Figure 6.3.5** our average line is at 2.0 which is quite good compare to our benchmark baseline model. However, at this point we not sure if this model is the best strategy until we compare it with another model.

6.4 Phase II Result

6.4.1 Hyperparameter Search & Training the Model

MLP Hyperparameter Tuning	Total Return	Annual Volatility	Accuracy	Sharpe Ratio
HL(128), Relu, Adam, epoch (500)*	3.02	1.11	0.4230	0.71
HL(100), Relu, Adam, epoch(500), LR(0.001)	3.7624	1.09	0.4602	0.76
HL(128),Relu, Adam, epoch(500), LR(0.01),alpha(0.001)	5.5782	0.95	0.4764	0.77

*Default parameter

Table 6.4.1 Multi-Layer Perceptron Hyperparameter Optimizing

Random Forest Hyperparameter Tuning	Total Return	Annual Volatility	Accuracy	Sharpe Ratio
n_estimator (100), max depth(none)*	5.43	1.09	0.48	0.51
n_estimator (200), max depth (none)	4.09	1.09	0.48	0.58
n_estimator (200), max depth (10)	5.43	1.09	0.48	0.79

*Default parameter

Table 6.4.2 Random Forest Hyperparameter Optimizing

For our phase II experiment, we start by finding best hyperparameter for 2 of our machine learning model, the random forest and multi-layer perceptron. For each algorithm, we try 3 combination and one of them is the default parameter set by Sklearn library. Our objective in hyperparameter tuning is to find the best combination of parameter that produce performance higher than the default parameter performance. As per **Table 6.4.1**, our best hyperparameter is combination of Hidden Layer i.e. HL (128 nodes) with Relu activation function and Adam optimizer, maximum iteration i.e. epoch (500), learning rate i.e. LR (0.01) and finally our L2

regularization i.e. alpha at 0.001. This combination give us total return of 5.57% and Sharpe ratio of 0.77, an improvement of 0.06 from the default parameter.

In Random Forest optimization, our best combination was estimator (200) and maximum depth(10) that give us 5.43 total return and improvement in Sharpe Ratio of 0.79% compare to default parameter that only give around 0.51%.

```
Total Returns:
returns                  1.287017
strategy_log_reg         1.744055
strategy_gauss_nb        0.911785
strategy_svm             3.533975
strategy_random_forest   5.321431
strategy_MLP             7.558861
dtype: float64

Annual Volatility:
returns                  1.110451
strategy_log_reg         1.094012
strategy_gauss_nb        0.964071
strategy_svm             1.093622
strategy_random_forest   1.093258
strategy_MLP             1.092865
dtype: float64
```

Figure 6.4 Summary Result All Model

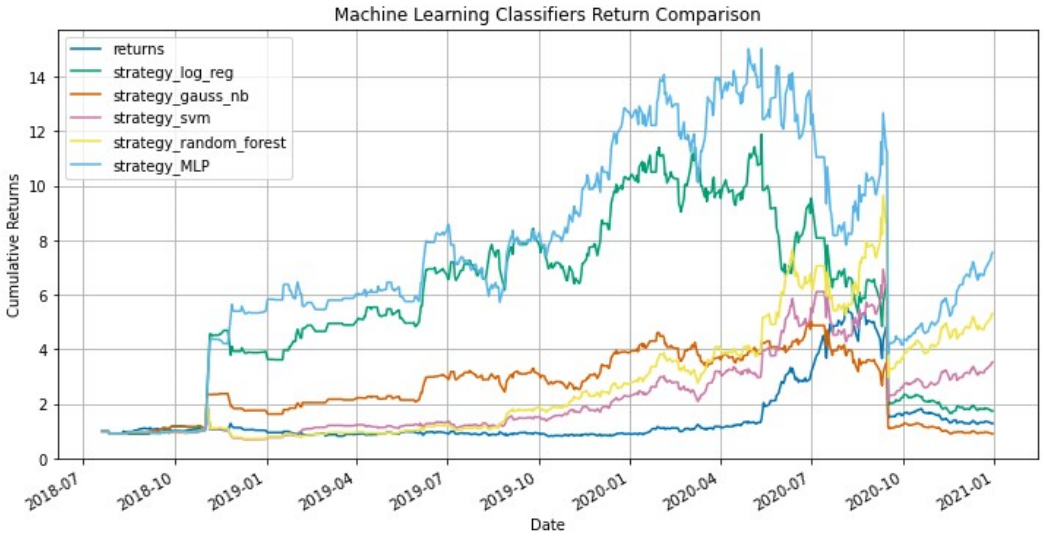


Figure 6.4.1 Performance graph on all model

```
# number of trades over time top 3 highest return strategy
print('Number of trades SVM = ', (stock['pos_svm'].diff()!=0).sum())
print('Number of trades Multilayer Perceptron = ',(stock['pos_MLP'].diff()!=0).sum())
print('Number of trades Random Forest = ',(stock['pos_random_forest'].diff()!=0).sum())
```

```
Number of trades SVM =  226
Number of trades Multilayer Perceptron =  275
Number of trades Random Forest =  238
```

Figure 6.4.2 Number of Transaction

Then after that, we train all 5 model and find the performance of each model and the result as shown in **Figure 6.4** and visualization are in **Figure 6.4.1.** The goal in this step is to find top 3 model that we will selected to be test in backtesting platform. From **Figure 6.4.2,** MLP recorded highest number of transactions with 275 trades while Random Forest is 238 and SVM 226 trades.

6.4.2 Evaluation

	Logistic Regression	Gaussian Naive Bayes	Support Vector Machine	Random Forest	Multi-Layer Perceptron
Accuracy Score	0.442464	0.424635	0.479741	0.482982	0.470016

Figure 6.4.3 Accuracy score for all model

Logistic Regression:

	precision	recall	f1-score	support
-1	0.40	0.39	0.39	230
0	0.49	0.28	0.35	141
1	0.46	0.59	0.52	246
accuracy			0.44	617
macro avg	0.45	0.42	0.42	617
weighted avg	0.44	0.44	0.43	617

Gaussian Naive Bayes:

	precision	recall	f1-score	support
-1	0.40	0.23	0.29	230
0	0.38	0.47	0.42	141
1	0.46	0.59	0.52	246
accuracy			0.42	617
macro avg	0.41	0.43	0.41	617
weighted avg	0.42	0.42	0.41	617

Support Vector Machine:

	precision	recall	f1-score	support
-1	0.50	0.38	0.43	230
0	0.49	0.28	0.35	141
1	0.47	0.69	0.56	246
accuracy			0.48	617
macro avg	0.49	0.45	0.45	617
weighted avg	0.48	0.48	0.46	617

Random Forest :

	precision	recall	f1-score	support
-1	0.50	0.37	0.43	230
0	0.49	0.28	0.35	141
1	0.47	0.70	0.57	246
accuracy			0.48	617
macro avg	0.49	0.45	0.45	617
weighted avg	0.49	0.48	0.47	617

Multilayer Perceptron:

	precision	recall	f1-score	support
-1	0.47	0.38	0.42	230
0	0.49	0.28	0.35	141
1	0.46	0.66	0.54	246
accuracy			0.47	617
macro avg	0.47	0.44	0.44	617
weighted avg	0.47	0.47	0.45	617

Figure 6.4.4 Precision recall evaluation metrics

For evaluation process of our model, the accuracy of each model was recorded as **Figure 6.4.3** and precision recall metrics was in **Figure 6.4.4.** The top accuracy between the 5 model is Random Forest with 0.4829 with precision of 0.49 (weighted average), recall at 0.48 and f1-score of 0.47 which is the highest among all the model. Second highest model are Support Vector Machine (SVM) with accuracy of 0.4797, precision of 0.48 (weighted average), recall of 0.48 and f1-score of 0.46. Third best model are Multi-Layer perceptron with accuracy (0.4700), precision (0.47), recall (0.47) and f1-score (0.45). Both logistic regression and gaussian naïve bayes score

lower than our top 3 model and will not be used in our backtesting platform to save the computation power and time.

6.4.3 Backtesting

SVM

Start date	2018-07-20
End date	2020-12-30
Total months	29
	Backtest
Annual return	22.3%
Cumulative returns	63.7%
Annual volatility	93.8%
Sharpe ratio	0.63
Calmar ratio	0.37
Stability	0.41
Max drawdown	-61.0%
Omega ratio	1.19
Sortino ratio	1.18
Skew	6.87
Kurtosis	134.18
Tail ratio	1.22
Daily value at risk	-11.6%

Random Forest

Start date	2018-07-20
End date	2020-12-30
Total months	29
	Backtest
Annual return	24.1%
Cumulative returns	69.6%
Annual volatility	98.1%
Sharpe ratio	0.72
Calmar ratio	0.33
Stability	0.45
Max drawdown	-73.3%
Omega ratio	1.24
Sortino ratio	1.16
Skew	4.36
Kurtosis	127.07
Tail ratio	1.20
Daily value at risk	-12.1%

MLP

Start date	2018-07-20
End date	2020-12-30
Total months	29
	Backtest
Annual return	42.6%
Cumulative returns	138.3%
Annual volatility	90.0%
Sharpe ratio	0.79
Calmar ratio	0.68
Stability	0.51
Max drawdown	-62.3%
Omega ratio	1.26
Sortino ratio	1.52
Skew	7.67
Kurtosis	157.51
Tail ratio	1.25
Daily value at risk	-11.1%

Figure 6.4.5 Tear Sheet for Top 3 Model

Best 3 top model was selected from evaluation to be run on backtesting platform to simulate on real dataset and validate the finding. When running the strategy with historical market data based on **Figure 6.4.5,** the best performance strategy is MLP model with annual return of 42.6% and cumulative returns are 138.3%. Volatility score is good with annual volatility at 90% and maximum drawdown at 62.3%. While risk adjusted return with Sharpe ratio of 0.79 and daily value at risk only at 11.1%.

Second best model is Random Forest model with return (annual return 24.1%, cumulative return 69.6%), volatility (annual volatility 98.1%, maximum drawdown 73.3%) and risk adjusted return (Sharpe ratio 0.72, daily value at risk 12.1%). Third best is Support Vector Machine model with return (annual return 22.3%, cumulative return 63.7%), volatility (annual volatility 93.8%, maximum drawdown 61.0%) and risk adjusted return (Sharpe ratio 0.63, daily value at risk 11.6%).

Both MLP and Random Forest model performed better than the baseline benchmark model. Only SVM model perform worse than the benchmark. This show there is possible overfitting issued with SVM model as it's the only model that we not run

any hyperparameter optimizer before backtesting. Next, we visualize of all top 3 models' performance, analyze the finding, and compare between these top 3 model.

6.4.3.1 Comparation Between Top 3 Model Performance

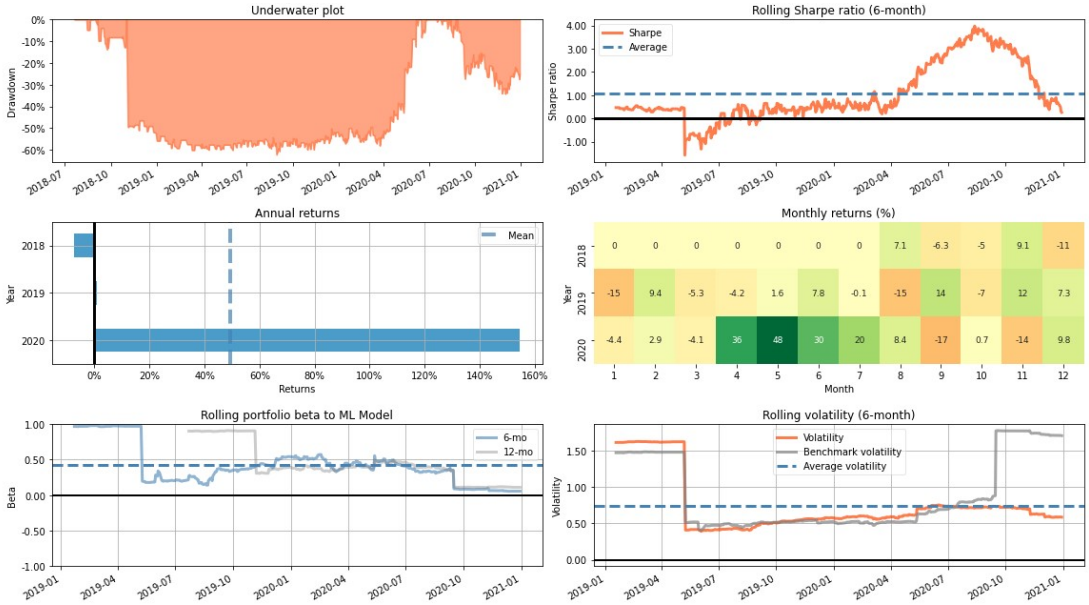


Figure 6.4.6 MLP Performance Chart

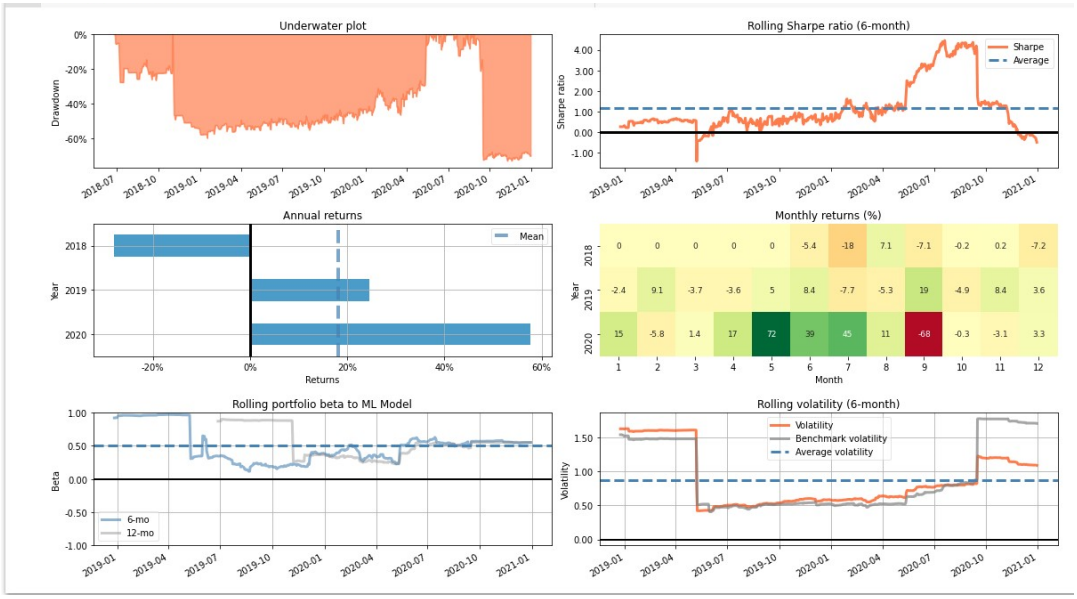


Figure 6.4.7 Random Forest Performance Chart

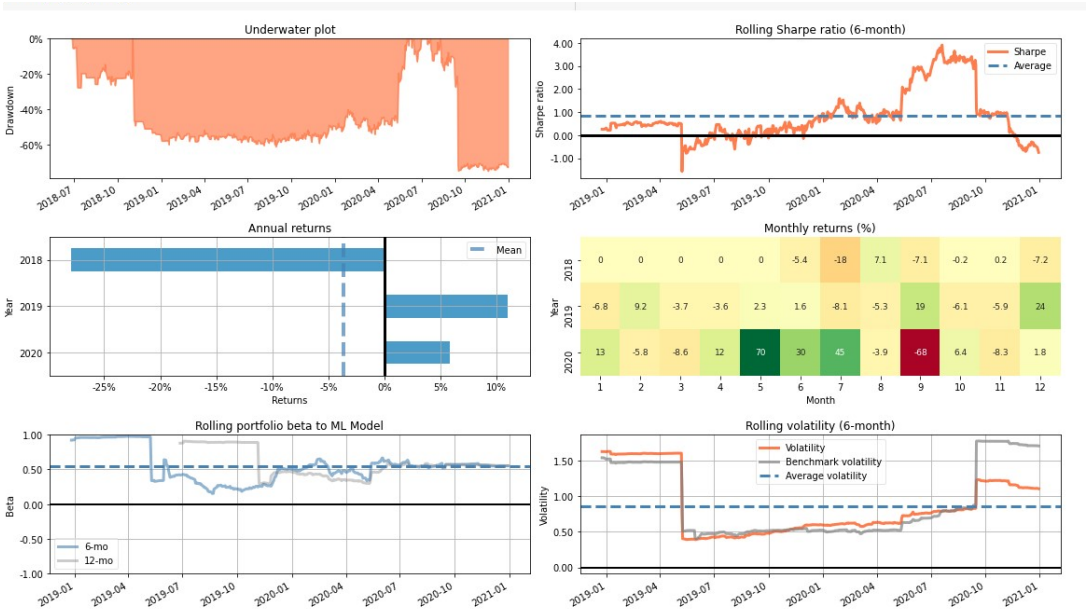


Figure 6.4.8 SVM Performance chart

In this section, the visualization of all 3-model performance and each models show 6 plotted chart (Underwater chart, Rolling Sharpe ratio, Annual returns, Monthly returns, Rolling portfolio beta to ML Model and Rolling volatility). Underwater plot is the visualization of the maximum drawdown (largest losses of particular stock), 6 month rolling Sharpe ratio are the Sharpe ratio with specific time frame, annual return of each year in observed time frame, monthly return is the monthly breakdown of return each year, rolling portfolio beta is the exposure of the strategy to broader market and rolling volatility is annual volatility with specific time frame.

When compare between these 3 models, underwater plot pinpoint how much drawdown and how long it takes for each strategy to recover from lowest point. MLP Model (**Figure 6.4.6**) show the worst drawdown occur around November 2018 and start to recover on April 2020 (18 month to recover) while both Random Forest and SVM (**Figure 6.4.7** & **Figure 6.4.8**) do recover from November 2018 event but hit even badly again on September 2020. This can be asserted in rolling volatility (6-month) chart that show only MLP model is under the benchmark for volatility while other are above the benchmark which. This proved that MLP model is robust during volatility period compared to another model we tested.

On return metrics, the rolling Sharpe ratio show that MLP model maintain above average line from April 2020 until November 2020 which is 7 months above average while Random Forest and SVM only last 4 month (May 2020 until September 2020)

above average line. The annual return mean shows that MLP model and Random Forest model are positive (MLP 42%, Random Forest 24%) while SVM annual return mean is negative value (-0.4 %). The monthly breakdown of return can be seen in Monthly Returns chart where highest return was on May 2020 and worst return is in September 2020 for all models. Again, this proved that MLP model is more stable than other models.

6.5 Overall Result

	Return		Volatility		Risk		
Candidate Model	Annual Return (%)	Cumulative Return (%)	Annual Volatility (%)	Maximum Drawdown (%)	Sharpe Ratio	Rolling Sharpe Ratio Avg	Daily Value at Risk (%)
SMA with SA	26.2	77.9	59.6	71.2	0.80	2.00	7.3
MLP	42.6	138.3	90.0	62.3	0.79	1.10	11.1
RF	24.1	69.6	98.1	73.3	0.72	1.10	12.1
B&H*	15.6	43.2	103.1	77.3	0.68	1.00	12.7
SVM	22.3	63.7	93.8	61	0.63	0.90	11.6

*Benchmark

Legend	
SMA	*Simple Moving Average*
SA	*Sentiment Analysis*
RF	*Random Forest*
B & H	*Buy & Hold*
SVM	*Support Vector Machine*

Table 6.5 Overall Performance of the Candidate Model

Table 6.5 show the compilation of overall result of our candidate strategy sorted by highest Sharpe Ratio result. From the table the best performance is the hybrid SMA + sentiment analysis followed by MLP and Random Forest. All of this model is above the benchmarks model.

When compared between SMA+ Sentiment Analysis model with MLP, there is huge different in term of return and volatility between both models. MLP have higher return compared to SMA+ Sentiment Analysis while in term of volatility, SMA+ Sentiment Analysis is less volatile compare to MLP. This can be explained by the number of transactions involved during the period as SMA+ Sentiment Analysis is low (<10 trades) compare to MLP (>200 trades) which in result expose the model with risk more often. This mean MLP model generate more often trading signal (refer to **Figure 6.4.3)** thus exposing the model to open position more while SMA + Sentiment Analysis only trades when there is positive news and crossover of moving average line which cause less in generating a trading signal. Another reason for this is the moving average that model used. SMA + Sentiment Analysis using 27 moving average which cause the model to become less sensitive to price changes.

Another point to highlight is that the SMA + Sentiment Analysis is suitable for long term investing while MLP is suitable for short term investment strategy. **Figure 6.3.5** show that SMA + Sentiment Analysis did not make any transaction during 2018 and 2019 due to the stock price did not move much during that period. However, in 2020 there is big movement of price which trigger the signal generated by this model. As discussed before, MLP model is more sensitive to small price movement thus making it suitable for short term investment.

6.6 Comparation With Previous Works

Researcher	Model Type	Market	Return (%)	Volatily (%)	Risk (Sharpe Ratio)
Xiong et al., 2018	Deep Reinforment Learning	DJIA (US)	25.87	N/a	1.79
Johnman, Vanstone & Gepp, 2018	Sentiment Analysis	FTSE (AUS)	6.00	169.6	0.33
This Study (2021)	Sentiment Analysis	KLSE (MAS)	26.20	59.6	0.82

Table 6.6 Previous Work Comparation

Based on **Table 6.6,** The result of this experiment was compared to previous researcher within the same field and domain. In research made by Columbia University, USA, researcher (Xiong et al., 2018) use state of the art, deep reinforcement learning to develop optimal trading strategy and compared it with Dow Jones Industrial Index (DJIA) in 2018 market condition. Their result was compared with DJIA and min variance portfolio that they created. The annual return of their strategy is 25.87 % while their Sharpe ratio is 1.79. Deep Reinforcement Learning (DRL) is considered state of the art of trading strategy. (Bacoyannis et al., 2018) stated that DRL can make 3600 decision per hour equivalence to 1 decision per second that show how high is the performance of DRL can be when applying in algorithmic trading. However, the drawback is DRL required high computation power and complexity that often not accessible to general public.

Another researcher form Bond University, Australia, researcher (Johnman, Vanstone & Gemp, 2018) develop their trading strategy using sentiment analysis and compare it with buy and hold strategy in FTSE 100 Australia also in 2018 market condition. The result they archive only 6% annual return with volatility 169.6% and 0.33 in Sharpe Ratio. While using Sentiment Analysis model proved to be weak learner individually but when pooled together with other statistical model, it able to generate signal that can improve more returns and better risk adjusted return as shown in our experiment.

6.7 Summary

In summary, this chapter show the finding and performance of each of candidate trading strategy that has been developed. The performance of developed strategy was evaluate and critically analyzed and then compare with each other. Then each model performance was back tested and visualize for validation and finally been compared with previous works done by researcher.

Chapter 7
Discussion and Conclusion

7.1 Introduction

In this chapter, the overall experiment will be summarized with each phase of the study will be discussed briefly. Conclusion are formed based on the result that we had obtained on the result and comparison with the objective stated earlier is achievable or not. The importance of this experiment and its contribution will be discussed later with future recommendation will be provided to improve the contribution to the fields.

7.2 Discussion and Conclusion

This experiment collected its datasets from 2 source namely financial news headline using web scraping technique and Yahoo finance API data that provide historical market data. The observation period of this experiment was during mid-2018 until late-2020 (29 month). The preprocessing was done separately due to different data type between both datasets. Financial news headline was treated with text preprocessing technique such as parsing, tokenization and applying sentiment score as input to our model while historical market data was treated with univariate time series data preprocessing such as data transformation and feature engineering. Major issue during the preprocessing was there is significant different score generated when using different lexicon in sentiment analysis where we experiment Vader lexicon and LM lexicon. After several experiment done, Loughran-McDonald finance lexicon turn out to give better result for our sentiment analysis.

After the data was prepared and transform, 6 model was constructed to be train and evaluate. 5 machine learning model and one statistical model was build and then their performance was evaluated and validate using backtesting platform to simulate real world data. SMA was the statistical model that we choose because the nature of stock market data that often complex, noisy and non-stationary, SMA able to smoothen the line by using moving average so the line will show less up and down. However, SMA is known for its drawback such as its only generate signal from past performance which has high variance and bias, therefore we combine with sentiment score which limit the variance and lowering the bias thus making the model more robust and versatile. In the development of the model, only three model was applied with hyperparameter tuning to increase the individual performance, namely SMA with objective to find optimum lag return that can produce

maximum return while 2 of our machines learning model (MLP and Random Forest) was run with hyperparameter search to find best accuracy with highest return.

The finding from result was then analyzed for each model. This result answers our previous research question that was thrown earlier. For example, we have presented 6 model to be implement by retails trader and our best performing model was SMA + Sentiment Analysis and followed by MLP model. Both of this model can be used as our trading strategy as there is not much different in term of Sharpe Ratio. MLP model is risky strategy but come with higher returns while SMA + Sentiment Analysis is less risky but still higher than benchmark model, the buy and hold strategy. Another key finding that algorithmic trading is more profitable than Buy and Hold Strategy in range from 10% to 27% depend on which strategy that the retail investor applies. Even though Deep learning is state of the art at this moment, but the cost of computation is high, and complexity mean that its not accessible to retail trader therefore the 6 model we presented is suitable to be used in Malaysian market with the option to use our hybrid SMA + Sentiment Analysis which is the highest performance among all the model.

Finally, the result was compared with previous researcher, and we found that the highest result was achieved by deep reinforcement learning with the Sharpe ratio of 1.73. Our model achieve 0.80 for SMA + Sentiment Analysis and 0.79 for our MLP model. And the researcher from Bond university only achieved 0.33 Sharpe ratio. It was established that 3 of our develop model was better than our benchmark model. Higher Sharpe ratio was achieved in this experiment and not far away from DLR thus its more feasible and can be used by retail investors.

In conclusion, we can conclude that we achieved all our research objective such as to explore several algorithm strategies in which we have develop 6 algorithm model to be used in algorithmic trading. We also have identify high performance trading strategy such as SMA + sentiment analysis model and MLP model that have high Sharpe ratio among our developed model and finally we recommended to use sentiment analysis score incorporate with SMA moving average to enhance the model capability in generating trade signal.

7.3 Important and Contribution of the Study

One of key contribution of this study is to show that it's possible to democratize algorithmic trading to the general public rather than previously only be used by institution traders. We have shown the source of free data together with 6 algorithms to be used and all of the algorithm doesn't require much computation power unlike deep learning based algorithm such as deep reinforcement learning.

Another key contribution is our hybrid model the SMA + Sentiment analysis develop model. This is enhanced version of SMA model that incorporate sentiment analysis score. This experiment proved that sentiment analysis model can be used together with other model to improve the robustness and accuracy thus increase the performance therefore it should not be use as single model. Retail investor can take advantage from this model whenever the market facing black swan event which generally increase the volatility of the current market condition. Most models failed when facing unexpected event but model with high robustness will endure high volatile market as shown in our model during the experiment.

7.4 Future Recommendation

For future recommendation, we would like to see the performance of sentiment analysis to be incorporate with another advance statistical model. Linear quadratic estimation (LQE) or also known as Kalman Filter is an algorithm that able to contain statistical noise and other inaccuracy such as outlier. Rather than using fixed size windows like our SMA, it uses probabilistic model to estimate current value of time series. By using the same framework as our SMA + Sentiment Analysis hybrid model, we believe Kalman filter has the potential to provide higher performance when combine with sentiment analysis than our model thus this application should be research in the future.

Reference

Abbasi, E., Samavi, M.E. and Koosha, E., 2020. Performance Evaluation of the Technical Analysis Indicators in Comparison whit the Buy and Hold Strategy in Tehran Stock Exchange Indices. Advances in Mathematical Finance and Applications, 5(3), pp.1-19.

Akhtar, M.S., Kumar, A., Ghosal, D., Ekbal, A. and Bhattacharyya, P., 2017, September. 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* (pp. 540-546).

Al-Jaifi, H.A., Al-rassas, A.H. and Al-Qadasi, A.A., 2017. Corporate governance strength and stock market liquidity in Malaysia. *International Journal of Managerial Finance*.

Andersen, A.C. and Mikelsen, S., 2012. A Novel Algorithmic Trading Framework Applying Evolution and Machine Learning for Portfolio Optimization. *Department of Industrial Economics and Technology Management*.

Angel, J.J., Harris, L.E. and Spatt, C.S., 2011. Equity trading in the 21st century. *The Quarterly Journal of Finance*, *1*(01), pp.1-53.

Bacoyannis, V., Glukhov, V., Jin, T., Kochems, J., & Song, D. R. (2018). Idiosyncrasies and challenges of data driven learning in electronic trading. arXiv preprint arXiv:1811.09549.

Borges, M.R., 2010. Efficient market hypothesis in European stock markets. *The European Journal of Finance*, *16*(7), pp.711-726

Bouazizi, M. and Ohtsuki, T.O., 2016. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, *4*, pp.5477-5488.

Cohen, G. and Cabiri, E., 2015. Can technical oscillators outperform the buy and hold strategy?. *Applied Economics*, *47*(30), pp.3189-3197.

Cohen, G. and Cabiri, E., 2015. Can technical oscillators outperform the buy and hold strategy?. Applied Economics, 47(30), pp.3189-3197.

Dabrowski, M., 2017. Potential impact of financial innovation on financial services and monetary policy. *CASE Research Paper*, (488).

Daniel, K., Grinblatt, M., Titman, S. and Wermers, R., 1997. Measuring mutual fund performance with characteristic-based benchmarks. *The Journal of finance*, *52*(3), pp.1035-1058.

Dechow, P.M., Hutton, A.P., Meulbroek, L. and Sloan, R.G., 2001. Short-sellers, fundamental analysis, and stock returns. *Journal of financial Economics*, *61*(1), pp.77-106.

Dempster, M.A.H., Evstigneev, I.V. and Schenk-Hoppe, K.R., 2008. Financial markets. The joy of volatility. *Quantitative Finance*, *8*(1), pp.1-3.

Djuriš, J., Medarević, D., Krstić, M., Vasiljević, I., Mašić, I. and Ibrić, S., 2012. Design space approach in optimization of fluid bed granulation and tablets compression process. The Scientific World Journal, 2012.

Dong, X.L. and De Melo, G., 2018, July. A helping hand: Transfer learning for deep sentiment analysis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 2524-2534).

Eapen, J., Bein, D. and Verma, A., 2019, January. Novel deep learning model with CNN and bi-directional LSTM for improved stock market index prediction. In *2019 IEEE 9th annual computing and communication workshop and conference (CCWC)* (pp. 0264-0270). IEEE.

Fama, E.F., 1995. Random walks in stock market prices. *Financial analysts journal*, *51*(1), pp.75-80.

Gilbert, C.H.E. and Hutto, E., 2014, June. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) http://comp. social. gatech. edu/papers/icwsm14. vader. hutto. pdf* (Vol. 81, p. 82).

Holden, C.W. and Subrahmanyam, A., 2002. News events, information acquisition, and serial correlation. *The Journal of Business*, *75*(1), pp.1-32.

Hsieh, C.H., Barmish, B.R. and Gubner, J.A., 2019, December. The impact of execution delay on kelly-based stock trading: High-frequency versus buy and hold. In *2019 IEEE 58th Conference on Decision and Control (CDC)* (pp. 2580-2585). IEEE.

Hsieh, C.H., Barmish, B.R. and Gubner, J.A., 2019, December. The impact of execution delay on kelly-based stock trading: High-frequency versus buy and hold. In 2019 IEEE 58th Conference on Decision and Control (CDC) (pp. 2580-2585). IEEE.

Jing, X., Talekar, C. and Rayz, J.T., 2018, July. Comparing Jokes with NLP: How Far Can Joke Vectors Take Us?. In *International Conference on Distributed, Ambient, and Pervasive Interactions* (pp. 310-326). Springer, Cham.

Johnman, M., Vanstone, B.J. and Gepp, A., 2018. Predicting FTSE 100 returns and volatility using sentiment analysis. Accounting & Finance, 58, pp.253-274.

Karmiani, D., Kazi, R., Nambisan, A., Shah, A. and Kamble, V., 2019, February. Comparison of predictive algorithms: Backpropagation, svm, lstm and kalman filter for stock market. In *2019 Amity International Conference on Artificial Intelligence (AICAI)* (pp. 228-234). IEEE.

Kewat, P., Sharma, R., Singh, U. and Itare, R., 2017, April. Support vector machines through financial time series forecasting. In *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)* (Vol. 2, pp. 471-477). IEEE.

Kim, Y., Jeong, S.R. and Ghani, I., 2014. Text opinion mining to analyze news for stock market prediction. *Int. J. Advance. Soft Comput. Appl*, *6*(1), pp.2074-8523.

Lee, J.W., Park, J., Jangmin, O., Lee, J. and Hong, E., 2007. A multiagent approach to q-learning for daily stock trading. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *37*(6), pp.864-877.

Liu, Y. and Wu, Y.F.B., 2018, April. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Loh, E.Y., 2007. An alternative test for weak form efficiency based on technical analysis. *Applied Financial Economics*, *17*(12), pp.1003-1012.

Long, W., Lu, Z. and Cui, L., 2019. Deep learning-based feature engineering for stock price movement prediction. Knowledge-Based Systems, 164, pp.163-173.

Loughran, T. and McDonald, B., 2016. Textual analysis in accounting and finance: A survey. Journal of Accounting Research, 54(4), pp.1187-1230.

Medhat, W., Hassan, A. and Korashy, H., 2014. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, *5*(4), pp.1093-1113.

Poria, S., Peng, H., Hussain, A., Howard, N. and Cambria, E., 2017. Ensemble application of convolutional neural networks and multiple kernel learning for multimodal sentiment analysis. *Neurocomputing*, *261*, pp.217-230.

Saad, E.W., Prokhorov, D.V. and Wunsch, D.C., 1996, June. Advanced neural network training methods for low false alarm stock trend prediction. In Proceedings of International Conference on Neural Networks (ICNN'96) (Vol. 4, pp. 2021-2026). IEEE.

Saragih, J.M., Lucey, S. and Cohn, J.F., 2009, September. Face alignment through subspace constrained mean-shifts. In *2009 IEEE 12th International Conference on Computer Vision* (pp. 1034-1041). Ieee.

Sharma, N. and Juneja, A., 2017, April. Combining of random forest estimates using LSboost for stock market index prediction. In *2017 2nd International Conference for Convergence in Technology (I2CT)* (pp. 1199-1202). IEEE.

Shilling, A.G., 1992. Market timing: Better than a buy-and-hold strategy. *Financial Analysts Journal*, *48*(2), pp.46-50.

Shynkevich, Y., McGinnity, T.M., Coleman, S. and Belatreche, A., 2015, July. Stock price prediction based on stock-specific and sub-industry-specific news articles. In *2015 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.

Smith, G. and Ryoo, H.J., 2003. Variance ratio tests of the random walk hypothesis for European emerging stock markets. *The European Journal of Finance*, *9*(3), pp.290-300

Sushko, V. and Turner, G., 2018. The implications of passive investing for securities markets. *BIS Quarterly Review, March*.

Tome, J.A.B. and Carvalho, J.P., 2005, November. Market index prediction using fuzzy boolean nets. In *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)* (pp. 5-pp). IEEE.

ur Rahman, W., 2019. Validity of Random Walk Hypothesis and Technical Analysis: An Investigation of Pakistan Stock Market. *Global Management Journal for Academic & Corporate Studies*, *9*(1), pp.100-120.

Usmani, M., Adil, S.H., Raza, K. and Ali, S.S.A., 2016, August. Stock market prediction using machine learning techniques. In *2016 3rd international conference on computer and information sciences (ICCOINS)* (pp. 322-327). IEEE.

Xiong, Z., Liu, X.Y., Zhong, S., Yang, H. and Walid, A., 2018. Practical deep reinforcement learning approach for stock trading. arXiv preprint arXiv:1811.07522.

Yam, S.C.P., Yung, S.P. and Zhou, W., 2009. Two rationales behind the 'buy-and-hold or sell-at-once'strategy. *Journal of Applied Probability*, *46*(3), pp.651-668.

Appendix A

Google Colab Code Accessible Here :-

(https://colab.research.google.com/drive/1qkunXEJGpgGo3uBtgYa7_VvyW--8GsSb)

Python Code:

```
# -*- coding: utf-8 -*-
"""capstoneproject

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1qkunXEJGpgGo3uBtgYa7_VvyW--8GsSb
"""

#install Library
!pip install backtrader
!pip install pyfolio
!pip install bs4 --user
!pip install yfinance
!pip install pysentiment2

"""# **Base Strategy - Buy and Hold Strategy**"""

import backtrader as bt


class BuyAndHold_Buy(bt.Strategy):
    def start(self):
        self.val_start = self.broker.get_cash()  # keep the starting cash

    def nextstart(self):
        # Buy stocks with all the available cash
        size = int(self.val_start / self.data)
        self.buy(size=size)

    def stop(self):
        # calculate the actual returns
        self.roi = (self.broker.get_value() / self.val_start) - 1.0
        print("ROI: %.2f, Cash: %.2f" % (100.0 * self.roi, self.broker.get_val
ue()))

from datetime import datetime
import pyfolio as pf


data = bt.feeds.YahooFinanceData(
```

```
    dataname="TGLVY", fromdate=datetime(2018, 7, 12), todate=datetime(2020, 12
, 31)
)


cerebro = bt.Cerebro()

cerebro.adddata(data)
cerebro.broker.setcash(100000.0)


cerebro.addstrategy(BuyAndHold_Buy, "HOLD")
# add analyzers
cerebro.addanalyzer(bt.analyzers.Returns, _name='returns')
cerebro.addanalyzer(bt.analyzers.TimeReturn, _name='time_return')
cerebro.addanalyzer(bt.analyzers.PyFolio, _name='pyfolio')

# Execute
print('Starting Portfolio Value: %.2f' % cerebro.broker.getvalue())
results = cerebro.run(stdstats=True, tradehistory=False)
print('Final Portfolio Value: %.2f' % cerebro.broker.getvalue())

# Extract inputs for pyfolio
strat = results[0]
pyfoliozer = strat.analyzers.getbyname('pyfolio')
# Extract inputs for pyfolio
returns, positions, transactions, gross_lev = pyfoliozer.get_pf_items()
returns.name = 'Strategy'
returns.head(2)

pf.show_perf_stats(returns,)

benchmark_rets= returns
benchmark_rets.index = benchmark_rets.index.tz_convert('UTC')
benchmark_rets = benchmark_rets.filter(returns.index)
benchmark_rets.name = 'BuyHold'
benchmark_rets.head(2)

# Commented out IPython magic to ensure Python compatibility.
# %matplotlib inline
import matplotlib.pyplot as plt

# silence warnings
import warnings
warnings.filterwarnings('ignore')

# plot performance for strategy vs benchmark
fig, ax = plt.subplots(nrows=3, ncols=2, figsize=(16, 9),constrained_layout=Tr
ue)
```

```
axes = ax.flatten()

axes[1].grid(True)
pf.plot_drawdown_underwater(returns=returns, ax=axes[0])

axes[2].grid(True)
pf.plot_rolling_sharpe(returns=returns, ax=axes[1])

pf.plot_annual_returns(returns=returns, ax=axes[2])
axes[2].grid(True)

pf.plot_monthly_returns_heatmap(returns=returns, ax=axes[3],)

pf.plot_rolling_beta(returns=returns, factor_returns=benchmark_rets, ax=axes[4
])
axes[4].grid(True)

pf.plot_rolling_volatility(returns=returns, factor_returns=benchmark_rets,ax=a
xes[5])
axes[5].grid(True)

plt.tight_layout()

# Commented out IPython magic to ensure Python compatibility.
#Force reset to clear all input
# %reset -f

"""# **Phase I - Sentiment Analysis**"""

import nltk
import warnings
warnings.filterwarnings('ignore')
import pysentiment2 as ps
import urllib
from urllib.request import urlopen
from urllib.error import HTTPError
from bs4 import BeautifulSoup
from datetime import datetime, timedelta
import time
import pprint

#Using Loughran and McDonald dictionary as main financial corpus
LNM = ps.LM()

#Extract news headline and score their sentiment using sentiment analyzer
date_sentiments = {}
for i in range (1,11):
```

```
    page = urlopen('https://www.businesstimes.com.sg/search/top+glove?page='+s
tr(i)).read()
    soup = BeautifulSoup(page, features="html.parser")
    posts = soup.findAll("div", {"class": "media-body"})# find the header
    for post in posts:
        time.sleep(2)
        url = post.a['href']
        date = post.time.text
        print(date, url)
        for setr in posts:
            passage=setr.find("p").text #save all the header in passage
        #use loughran mcdonald to calculate polarity of positive|negative news
        sentiment1=LNM.tokenize(passage)
        sentiment=LNM.get_score(sentiment1)['Polarity']
        date_sentiments.setdefault(date, []).append(sentiment)
date_sentiment = {}
for k,v in date_sentiments.items():
    date_sentiment[datetime.strptime(k, '%d %b %Y').date() + timedelta(days=1)
] = round(sum(v)/float(len(v)),3)

earliest_date = min(date_sentiment.keys())

print(date_sentiments)

#save the result in pickle so that we dont have to scrape the everytime we run
 the program

from google.colab import drive
from google.colab import drive
drive.mount('/content/drive')

# import pickle
# with open('/content/drive/MyDrive/save_webscrape', 'wb') as f:
#     pickle.dump(date_sentiment, f)
# with open('/content/drive/MyDrive/save_webscrape1', 'wb') as f:
#     pickle.dump(date_sentiments, f)
# with open('/content/drive/MyDrive/save_webscrape2', 'wb') as f:
    # pickle.dump(earliest_date, f)
# #load the pickle object
import pickle
with open('/content/drive/MyDrive/save_webscrape', 'rb') as f:
    date_sentiment = pickle.load(f)
with open('/content/drive/MyDrive/save_webscrape1', 'rb') as f:
    date_sentiments = pickle.load(f)
with open('/content/drive/MyDrive/save_webscrape2', 'rb') as f:
    earliest_date = pickle.load(f)

# Commented out IPython magic to ensure Python compatibility.
```

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.cm as cm
from matplotlib.pyplot import figure
from matplotlib import rcParams


# %matplotlib inline
df= pd.DataFrame.from_dict(date_sentiments,orient= 'index', columns=['compound
','compound1', 'compound2','compound3','compound4'])
df.transpose()
#df = pd.DataFrame(df, columns=['Date', 'compound'])
df.index = pd.to_datetime(df.index)
df

#other compound are simply duplicate news within a day so we ignore it due to
simplicity

#Plotting the polarity score
plt.figure(figsize=(10,5))
plt.hist(df['compound'],label=['Score','Count'], align='left', rwidth=0.9, col
or='tab:orange', histtype='stepfilled')
plt.title('Sentiment Score On Each Top Glove News')
plt.xlabel('Polarity Score')
plt.ylabel('Count of Score')
plt.show()

"""# **Build Simple Moving Average Model**"""

#Run Hyperparameter Search to find best lagged for our Simple Moving Average

import yfinance
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import ttest_ind
import sklearn
from sklearn.metrics import mean_squared_error
import math
from datetime import datetime, timedelta

plt.rcParams['figure.figsize'] = [10, 7]
plt.rc('font', size=14)
np.random.seed(0)

y = np.arange(0,100,1) + np.random.normal(0,10,100)
sma = pd.Series(y).rolling(5).mean()
plt.plot(y,label="Time series")
plt.plot(sma,label="5-period SMA")
```

```
plt.legend()
plt.show()

#Walk Forward Prediction
n_forward = 5
name = 'TGLVY'
start_date = "2018-07-12"
end_date = "2020-12-31"

ticker = yfinance.Ticker('TGLVY')
data = ticker.history(interval="1d",start='2018-07-12',end=end_date)

#Visualize with common lagged period i.e. 5(Day Trader), 15(Swing Trader), 200
 (Long Trader)
plt.figure(figsize=(15,5))
plt.plot(data['Close'],label="Top Glove")

plt.plot(data['Close'].rolling(5).mean(),label = "5-periods SMA")
plt.plot(data['Close'].rolling(15).mean(),label = "15-periods SMA")
plt.plot(data['Close'].rolling(200).mean(),label = "200-periods SMA")

plt.legend()
plt.ylim((0,40))
plt.show()

ticker = yfinance.Ticker(name)
data = ticker.history(interval="1d",start=start_date,end=end_date)

data['Forward Close'] = data['Close'].shift(-n_forward)

data['Forward Return'] = (data['Forward Close'] - data['Close'])/data['Close']

result = []
train_size = 0.75 #split dataset into Train(75) & Test(25)

for sma_length in range(5,500):

  data['SMA'] = data['Close'].rolling(sma_length).mean()
  data['input'] = [int(x) for x in data['Close'] > data['SMA']]

  df = data.dropna()

  training = df.head(int(train_size * df.shape[0]))
  test = df.tail(int((1 - train_size) * df.shape[0]))

  tr_returns = training[training['input'] == 1]['Forward Return']
  test_returns = test[test['input'] == 1]['Forward Return']
```

```
  mean_forward_return_training = tr_returns.mean()
  mean_forward_return_test = test_returns.mean()

  pvalue = ttest_ind(tr_returns,test_returns,equal_var=False)[1]
  mse = sklearn.metrics.mean_squared_error(test['Close'], test['Forward Close'
])
  rmse = math.sqrt(mse)

  result.append({
      'sma_length':sma_length,
      'training_forward_return': mean_forward_return_training,
      'test_forward_return': mean_forward_return_test,
      'RMSE' : rmse,
      'MSE' : mse
  })

result.sort(key = lambda x : -x['test_forward_return'])
result[0]

#result
#n_forward(5) = [test_forward_return: 0.053, sma_length: 27, MSE: 9.3, RMSE: 3
.05]
#n_forward(10) = [test_forward_return: 0.112, sma_length: 25, MSE: 20.38, RMSE
: 4.51  ]
#n_forward(15) = [test_forward_return: 0.173, sma_length: 25, MSE: 33.69, RMSE
: 5.80 ]

best_sma = result[0]['sma_length']
data['SMA'] = data['Close'].rolling(best_sma).mean()

plt.figure(figsize=(15,5))
plt.plot(data['Close'],label=name)
plt.plot(data['SMA'],label = "{} periods SMA".format(best_sma))
plt.legend()
plt.show()

#Visualize the SMA with Sentiment Analysis as Trade Signal
data['BEST_MA'] = data['Close'].rolling(25).mean()
data.dropna(inplace = True)

#Combine Sentiment Analysis dataframe with SMA Dataframe
dfs= pd.DataFrame.from_dict(date_sentiments,orient= 'index', columns=['compoun
d','compound1', 'compound2','compound3','compound4'])
datax=data.join(pd.DataFrame(dfs).rename(columns={'compound':'SIA','compound1'
:'SIA1','compound2':'SIA2','compound3':'SIA3','compound4':'SIA4'}))

#MA Trade Calls
```

```
Trade_Buy = []
Trade_Sell = []

for i in range(len(datax) - 1):
    if ((datax['BEST_MA'].values[i] < datax['Close'].values[i]) & (datax['SIA'
].values[i+1] > 0.5)):
        Trade_Buy.append(i)
    elif ((datax['BEST_MA'].values[i] > datax['Close'].values[i]) & (datax['SI
A'].values[i+1] < 0)):
        Trade_Sell.append(i)

from datetime import timedelta
fig, ax = plt.subplots(1, 1, figsize = (20, 10))

ax.plot(datax.index, datax['Close'])
ax.plot(datax.index, datax['BEST_MA'],'-
^', markevery=Trade_Buy, ms=15, color = 'g')
ax.plot(data.index, datax['BEST_MA'],'-
v', markevery=Trade_Sell, ms=15, color = 'r')
ax.set_xlabel('Date', fontsize = 15)
ax.set_ylabel('Stock Price', fontsize = 15)
ax.tick_params(axis = 'x', labelsize = 13)
ax.tick_params(axis = 'y', labelsize = 13)
ax.set_title('Trade Signal - Moving Averages Algorithm', fontdict = {'size': 2
0})
ax.legend(['Close','Buy','Sell'])
ax.grid()

plt.subplots_adjust(top = 0.92, left = 0.09, right = 0.93, bottom = 0.14, hspa
ce = 0.3)
plt.show()

"""## **Run backtesting using SMA Optimize period**"""

from __future__ import (absolute_import, division, print_function,unicode_lite
rals)

import backtrader as bt
import backtrader.indicators as btind
import pyfolio as pf
import datetime
import os.path
import sys
import IPython

class Sentiment(bt.Indicator):
    lines = ('sentiment',)
    plotinfo = dict(
```

```
        plotmargin = 0.15,
        plothlines = [0],
        plotyticks = [1.0, 0, -1.0]
    )

    def next(self):
        self.date = self.data.datetime
        date = bt.num2date(self.date[0]).date()
        prev_sentiment = self.sentiment
        if date in date_sentiment:
            self.sentiment = date_sentiment[date]
            self.lines.sentiment[0] = self.sentiment

class SentimentStrat(bt.Strategy):
    params = (
        ('period', 27), #use optimized SMA Period
        ('printlog', True),
    )

    def log(self, txt, dt=None, doprint=False):
        ''' Logging function for this strategy'''
        if self.params.printlog or doprint:
            dt = dt or self.datas[0].datetime.date(0)
            print('%s, %s' % (dt.isoformat(), txt))

    def __init__(self):
        # Keep a reference to the "close" line in the data[0] dataseries
        self.dataclose = self.datas[0].close
        # Keep track of pending orders
        self.order = None
        self.buyprice = None
        self.buycomm = None
        self.sma = bt.indicators.SimpleMovingAverage(
            self.datas[0], period=self.params.period)
        self.date = self.data.datetime
        self.sentiment = None
        Sentiment(self.data)

    def notify_order(self, order):
        # Buy/Sell order submitted/accepted to/by broker -> nothing to do
        if order.status in [order.Submitted, order.Accepted]:
            return

        # Check if an order has been completed
        # Broker can reject order if not enough cash
        if order.status in [order.Completed]:
            if order.isbuy():
                self.log(
```

```
                    'BUY EXECUTED, Price: %.2f, Cost: %.2f, Comm %.2f'%
                    (order.executed.price,
                     order.executed.value,
                     order.executed.comm))
                self.buyprice = order.executed.price
                self.buycomm = order.executed.comm

            else:
                self.log('SELL EXECUTED, Price: %.2f, Cost: %.2f, Comm %.2f' %
                         (order.executed.price,
                          order.executed.value,
                          order.executed.comm))
                self.bar_executed = len(self)

        elif order.status in [order.Canceled, order.Margin, order.Rejected]:
            self.log('Order Canceled/Margin/Rejected')

        # No pending order
        self.order = None

    def notify_trade(self, trade):
        if not trade.isclosed:
            return
        self.log('OPERATION PROFIT, GROSS %.2f, NET %.2f'%(trade.pnl, trade.pn
lcomm))

    # Main strat
    def next(self):
        # log close price of the series from the reference
        self.log('Close, %.2f' % self.dataclose[0])

        date = bt.num2date(self.date[0]).date()
        prev_sentiment = self.sentiment
        if date in date_sentiment:
            self.sentiment = date_sentiment[date]

        # check if an order is pending, if yes, we can't send a 2nd one
        if self.order:
            return
        print("Sentiment Score",self.sentiment)
        if not self.position and prev_sentiment:
        # buy maximum shares of the stock if the sentiment increases by 0.5 AN
D currenct close more than sma
            if self.dataclose[0] > prev_sentiment >= 0.5:#self.sma[0] and self
.sentiment -
                self.log('BUY CREATE, %.2f'% self.dataclose[0])
                self.order = self.buy()
```

```
        elif prev_sentiment:
            # or sell if it decreases by 0.5 AND currenct close less than sma
            if self.dataclose[0] < self.sma[0] and self.sentiment - prev_senti
ment <= -0.5:
                self.log('SELL CREATE, %.2f'% self.dataclose[0])
                self.order = self.sell()

    def stop(self):
        self.log('(MA Period %2d) Ending Value %.2f' %
             (self.params.period, self.broker.getvalue()), doprint=True)

if __name__ == '__main__':
    # https://www.backtrader.com/docu/strategy/
    cerebro = bt.Cerebro()

    # Strategy
    cerebro.addstrategy(SentimentStrat)

    # Data Feed
    data = bt.feeds.YahooFinanceData(
        dataname = 'TGLVY',
        fromdate = earliest_date,
        todate = datetime.datetime(2020,12,31),
        reverse = False
    )

    cerebro.adddata(data)

    cerebro.broker.setcash(100000.0)
    cerebro.addsizer(bt.sizers.FixedSize, stake=20000)
    cerebro.broker.setcommission(commission=0.001)

   # add analyzers
    cerebro.addanalyzer(bt.analyzers.Returns, _name='returns')
    cerebro.addanalyzer(bt.analyzers.TimeReturn, _name='time_return')
    cerebro.addanalyzer(bt.analyzers.PyFolio, _name='pyfolio')



    print('Starting Portfolio Value: %.2f' % cerebro.broker.getvalue())
    results = cerebro.run(stdstats=True, tradehistory=False)
    print('Final Portfolio Value: %.2f' % cerebro.broker.getvalue())

import matplotlib
import matplotlib.pyplot as plt
import backtrader.analyzers as btanalyzers
import backtrader.feeds as btfeeds
from   backtrader.feeds import PandasData
import backtrader.strategies as btstrats
```

```
import backtrader.plot

# Extract inputs for pyfolio
strat = results[0]
pyfoliozer = strat.analyzers.getbyname('pyfolio')
# Extract inputs for pyfolio
returns, positions, transactions, gross_lev = pyfoliozer.get_pf_items()
returns.name = 'Strategy'
returns.head(2)

# get performance statistics for strategy
pf.show_perf_stats(returns,)

# get benchmark returns
benchmark_rets= returns
benchmark_rets.index = benchmark_rets.index.tz_convert('UTC')
benchmark_rets = benchmark_rets.filter(returns.index)
benchmark_rets.name = 'SMA'
benchmark_rets.head(2)

# Commented out IPython magic to ensure Python compatibility.
# %matplotlib inline
import matplotlib.pyplot as plt

# silence warnings
import warnings
warnings.filterwarnings('ignore')

# plot performance for strategy vs benchmark
fig, ax = plt.subplots(nrows=3, ncols=2, figsize=(16, 9),constrained_layout=Tr
ue)
axes = ax.flatten()

axes[1].grid(True)
pf.plot_drawdown_underwater(returns=returns, ax=axes[0])

axes[2].grid(True)
pf.plot_rolling_sharpe(returns=returns, ax=axes[1])

pf.plot_annual_returns(returns=returns, ax=axes[2])
axes[2].grid(True)

pf.plot_monthly_returns_heatmap(returns=returns, ax=axes[3],)

pf.plot_rolling_beta(returns=returns, factor_returns=benchmark_rets, ax=axes[4
])
axes[4].grid(True)
```

```
pf.plot_rolling_volatility(returns=returns, factor_returns=benchmark_rets,ax=a
xes[5])
axes[5].grid(True)

plt.tight_layout()

# Commented out IPython magic to ensure Python compatibility.
#Force reset to clear all input
# %reset -f

"""# **Phase II - Machine Learning Model**



"""

import numpy as np
from matplotlib import pyplot as plt
import pandas as pd
import seaborn as sns
import yfinance as yf
import warnings
import sklearn
from sklearn import linear_model
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import mean_squared_error
import math
import datetime
import pyfolio as pf
import backtrader as bt
from backtrader.feeds import PandasData



# set the style and ignore warnings
plt.style.use('seaborn-colorblind')
warnings.simplefilter(action='ignore', category=FutureWarning)
warnings.filterwarnings('ignore')

# ticker and the start and end dates for testing
ticker =  'TGLVY' # Top Glove ticker = TGLVY | Axiata = AXXTF | Maybank = MLYN
F
start = datetime.datetime(2018, 7, 12) #start date to extract
end = datetime.datetime(2020, 12, 31) #end date to extract

# Extract data from yahoo finance API
```

```
stock =  yf.download(ticker, progress=True, actions=True,start=start, end=end)
['Adj Close']
stock = pd.DataFrame(stock)#load into pandas dataframe
stock.rename(columns = {'Adj Close':ticker}, inplace=True)
stock.head(5)

# calculate daily log returns and market direction
stock['returns'] = np.log(stock / stock.shift(1))
stock.dropna(inplace=True)
stock['direction'] = np.sign(stock['returns']).astype(int)
stock.head(5)

# visualize the closing price and daily returns
fig, ax = plt.subplots(2, 1, sharex=True, figsize = (12,6))
ax[0].plot(stock[ticker], label = f'{ticker} Adj Close')
ax[0].set(title = f'{ticker} Closing Price', ylabel = 'Price')
ax[0].grid(True)
ax[0].legend()

ax[1].plot(stock['returns'], label = 'Daily Returns')
ax[1].set(title = f'{ticker} Daily Retunrs', ylabel = 'Returns')
ax[1].grid(True)
plt.legend()

#plt.tight_layout();
#plt.savefig('images\chart1', dpi=300) #used if want to save the image to loca
l machine

# define the number of lags
lags = [1, 2, 3, 4, 5]

# compute lagged log returns
cols = []
for lag in lags:
    col = f'rtn_lag{lag}'
    stock[col] = stock['returns'].shift(lag)
    cols.append(col)

stock.dropna(inplace=True)
stock.head(3)

# function to transform the lag returns to binary values (0,+1)
def create_bins(data, bins=[0]):
    global cols_bin
    cols_bin = []
    for col in cols:
        col_bin = col + '_bin'
        data[col_bin] = np.digitize(data[col], bins=bins)
```

```
        cols_bin.append(col_bin)

create_bins(stock)
stock[cols+cols_bin].head(10)

"""**Building the Top 3 Model**"""

# create a dictionary of selected algorithms & conduct manual hyperparameter o
ptimizer
models = {
    'log_reg': linear_model.LogisticRegression(),
    'gauss_nb': GaussianNB(),
    'svm': SVC(),
    'random_forest': RandomForestClassifier(n_estimators=200, max_depth=10),
    'MLP' : MLPClassifier(max_iter=500,hidden_layer_sizes=(128,),activation='r
elu',solver='adam',learning_rate_init=0.01, alpha=0.001),
}

# function that fits all models.
def fit_models(data):
    mfit = {model: models[model].fit(data[cols_bin], data['direction']) for mo
del in models.keys()}

# function that predicts (derives all position values) from the fitted models
def derive_positions(data):
    for model in models.keys():
        data['pos_' + model] = models[model].predict(data[cols_bin])

# function to evaluate all trading strategies
def evaluate_strats(data):
    global strategy_rtn
    strategy_rtn = []
    for model in models.keys():
        col = 'strategy_' + model
        data[col] = data['pos_' + model] * data['returns']
        strategy_rtn.append(col)
    strategy_rtn.insert(0, 'returns')

# fit the models
fit_models(stock)

# derives all position values
derive_positions(stock)

# evaluate all trading strategies by multiplying predicted directions to actua
l daily returns
evaluate_strats(stock)
```

```
# calculate total return and std. deviation of each strategy
print('\nTotal Returns:')
print(stock[strategy_rtn].sum().apply(np.exp))
print('\nAnnual Volatility:')
stock[strategy_rtn].std() * 252 ** 0.5

from sklearn.metrics import accuracy_score
Accuracy = [accuracy_score(stock['direction'], stock['pos_log_reg']), accuracy
_score(stock['direction'], stock['pos_gauss_nb']),accuracy_score(stock['direct
ion'], stock['pos_svm']), accuracy_score(stock['direction'], stock['pos_random
_forest']),accuracy_score(stock['direction'], stock['pos_MLP'])]

#store in dataframe
pd.DataFrame([Accuracy],index=['Accuracy Score'],columns=['Logistic Regression
','Gaussian Naive Bayes', 'Support Vector Machine','Random Forest','Multi-
Layer Perceptron '])

from sklearn.metrics import classification_report

print("Logistic Regression:\n",classification_report(stock['direction'], stock
['pos_log_reg']))
print("Gaussian Naive Bayes:\n",classification_report(stock['direction'], stoc
k['pos_gauss_nb']))
print("Support Vector Machine:\n",classification_report(stock['direction'], st
ock['pos_svm']))
print("Random Forest :\n",classification_report(stock['direction'], stock['pos
_random_forest']))
print("Multilayer Perceptron:\n",classification_report(stock['direction'], sto
ck['pos_MLP']))

# number of trades over time top 3 highest return strategy
print('Number of trades SVM = ', (stock['pos_svm'].diff()!=0).sum())
print('Number of trades Multilayer Perceptron = ',(stock['pos_MLP'].diff()!=0)
.sum())
print('Number of trades Random Forest = ',(stock['pos_random_forest'].diff()!=
0).sum())

# vectorized backtesting of the resulting trading strategies and visualize the
 performance over time
ax = stock[strategy_rtn].cumsum().apply(np.exp).plot(figsize=(12, 6),
                                                     title = 'Machine Learning
 Classifiers Return Comparison')
ax.set_ylabel("Cumulative Returns")
ax.grid(True);
# plt.tight_layout();
# plt.savefig('images/chart2', dpi=300)

"""## **Run Backtesting**"""
```

```
# fetch the daily pricing data from yahoo finance
prices =  yf.download(ticker, progress=True, actions=True, start=start, end=en
d)
prices.head(2)

# rename the columns as needed for Backtrader
prices.drop(['Close','Dividends','Stock Splits'], inplace=True, axis=1)
prices.rename(columns = {'Open':'open','High':'high','Low':'low','Adj Close':'
close','Volume':'volume',
                         }, inplace=True)
prices.head(3)

# add the predicted column to prices dataframe. This will be used as signal fo
r buy or sell
predictions = stock['strategy_MLP'] # top 3 = strategy_random_forest | strateg
y_svm | strategy_MLP
predictions = pd.DataFrame(predictions)
predictions.rename(columns = {'strategy_MLP':'predicted'}, inplace=True)# top
3 = strategy_random_forest | strategy_svm | strategy_MLP
prices = predictions.join(prices, how='right').dropna()
prices.head(5)

OHLCV = ['open', 'high', 'low', 'close', 'volume']

# class to define the columns we will provide
class SignalData(PandasData):
    """
    Define pandas DataFrame structure
    """
    cols = OHLCV + ['predicted']

    # create lines
    lines = tuple(cols)

    # define parameters
    params = {c: -1 for c in cols}
    params.update({'datetime': None})
    params = tuple(params.items())

# define backtesting strategy class
class MLStrategy(bt.Strategy):
    params = dict(
    )

    def __init__(self):
        # keep track of open, close prices and predicted value in the series
        self.data_predicted = self.datas[0].predicted
```

```
        self.data_open = self.datas[0].open
        self.data_close = self.datas[0].close

        # keep track of pending orders/buy price/buy commission
        self.order = None
        self.price = None
        self.comm = None

    # logging function
    def log(self, txt):
        '''Logging function'''
        dt = self.datas[0].datetime.date(0).isoformat()
        print(f'{dt}, {txt}')

    def notify_order(self, order):
        if order.status in [order.Submitted, order.Accepted]:
            # order already submitted/accepted - no action required
            return

        # report executed order
        if order.status in [order.Completed]:
            if order.isbuy():
                self.log(f'BUY EXECUTED --
- Price: {order.executed.price:.2f}, Cost: {order.executed.value:.2f},Commissi
on: {order.executed.comm:.2f}'
                )
                self.price = order.executed.price
                self.comm = order.executed.comm
            else:
                self.log(f'SELL EXECUTED --
- Price: {order.executed.price:.2f}, Cost: {order.executed.value:.2f},Commissi
on: {order.executed.comm:.2f}'
                )

        # report failed order
        elif order.status in [order.Canceled, order.Margin,
                              order.Rejected]:
            self.log('Order Failed')

        # set no pending order
        self.order = None

    def notify_trade(self, trade):
        if not trade.isclosed:
            return
        self.log(f'OPERATION RESULT --
- Gross: {trade.pnl:.2f}, Net: {trade.pnlcomm:.2f}')
```

```
    # We have set cheat_on_open = True.This means that we calculated the signa
ls on day t's close price,
    # but calculated the number of shares we wanted to buy based on day t+1's
open price.
    def next_open(self):
        if not self.position:
            if self.data_predicted > 0:
                # calculate the max number of shares ('all-in')
                size = int(self.broker.getcash() / self.datas[0].open)
                # buy order
                self.log(f'BUY CREATED --
- Size: {size}, Cash: {self.broker.getcash():.2f}, Open: {self.data_open[0]},
Close: {self.data_close[0]}')
                self.buy(size=size*0.99)
        else:
            if self.data_predicted < 0:
                # sell order
                self.log(f'SELL CREATED --- Size: {self.position.size}')
                self.sell(size=self.position.size)

# instantiate SignalData class
data = SignalData(dataname=prices)

# instantiate Cerebro, add strategy, data, initial cash, commission and pyfoli
o for performance analysis
cerebro = bt.Cerebro(stdstats = False, cheat_on_open=True)
cerebro.addstrategy(MLStrategy)
cerebro.adddata(data, name=ticker)
cerebro.broker.setcash(100000.0)
cerebro.broker.setcommission(commission=0.001)
cerebro.addanalyzer(bt.analyzers.PyFolio, _name='pyfolio')

# run the backtest
print('Starting Portfolio Value: %.2f' % cerebro.broker.getvalue())
backtest_result = cerebro.run()
print('Final Portfolio Value: %.2f' % cerebro.broker.getvalue())

#RF Final Portfolio Value: 337380.55
#SVM Final Portfolio Value: 231954.50
#MLP Final Portfolio Value: 113014.31

# Extract inputs for pyfolio
strat = backtest_result[0]
pyfoliozer = strat.analyzers.getbyname('pyfolio')
returns, positions, transactions, gross_lev = pyfoliozer.get_pf_items()
returns.name = 'Strategy'
returns.head(2)
```

```
# get benchmark returns
benchmark_rets= stock['returns']
benchmark_rets.index = benchmark_rets.index.tz_localize('UTC')
benchmark_rets = benchmark_rets.filter(returns.index)
benchmark_rets.name = 'ML Model'
benchmark_rets.head(2)

# get performance statistics for strategy
pf.show_perf_stats(returns)

# Commented out IPython magic to ensure Python compatibility.
# %matplotlib inline
import matplotlib.pyplot as plt

# silence warnings
import warnings
warnings.filterwarnings('ignore')

# plot performance for strategy vs benchmark
fig, ax = plt.subplots(nrows=3, ncols=2, figsize=(16, 9),constrained_layout=Tr
ue)
axes = ax.flatten()

axes[1].grid(True)
pf.plot_drawdown_underwater(returns=returns, ax=axes[0])

axes[2].grid(True)
pf.plot_rolling_sharpe(returns=returns, ax=axes[1])

pf.plot_annual_returns(returns=returns, ax=axes[2])
axes[2].grid(True)

pf.plot_monthly_returns_heatmap(returns=returns, ax=axes[3],)

pf.plot_rolling_beta(returns=returns, factor_returns=benchmark_rets, ax=axes[4
])
axes[4].grid(True)

pf.plot_rolling_volatility(returns=returns, factor_returns=benchmark_rets,ax=a
xes[5])
axes[5].grid(True)

plt.tight_layout()
```

Appendix B

ETHICAL APPROVAL OF RESEARCH PROJECT

Office Record	Receipt – Fast-Track Ethical Approval
Date Received:	Student name:Ahmad Fuad Khalit Student number:TP058497 Received by:
Received by whom:	Date:

APU / APIIT FAST-TRACK ETHICAL APPROVAL FORM (STUDENTS)

Tick one box (level of study):

☑ POSTGRADUATE (~~PhD / MPhil /~~ Masters)

☐ UNDERGRADUATE (Bachelors degree)

☐ FOUNDATION / DIPLOMA / Other categories

Tick one box (purpose of approval):

☐ Thesis / Dissertation / FYP project

☐ Module assignment

☑ Other: Capstone Project Proposal

Title of Programme on which enrolled : MSc in Data Science and Business Analytics

Tick one box: ☑ Full-Time Study or ☐ Part-Time Study

Title of project / assignment: Research Methodology for Capstone Project

Name of student researcher: Ahmad Fuad Khalit

Name of supervisor / lecturer: Dr.Waddah Waheeb Hassan Saeed

Student Researchers- please note that certain professional organisations have ethical guidelines that you may need to consult when completing this form.

Supervisors/Module Lecturers - please seek guidance from the Chair of the APU Research Ethics Committee if you are uncertain about any ethical issue arising from this application.

		YES	NO	N/A
1	Will you describe the main procedures to participants in advance, so that they are informed about what to expect?			✓
2	Will you tell participants that their participation is voluntary?			✓
3	Will you obtain written consent for participation?			✓
4	If the research is observational, will you ask participants for their consent to being observed?			✓
5	Will you tell participants that they may withdraw from the research at any time and for any reason?			✓
6	With questionnaires and interviews will you give participants the option of omitting questions they do not want to answer?			✓
7	Will you tell participants that their data will be treated with full confidentiality and that, if published, it will not be identifiable as theirs?			✓
8	Will you give participants the opportunity to be debriefed i.e. to find out more about the study and its results?			✓

If you have ticked **No** to any of Q1-8 you should complete the full Ethics Approval Form.

		YES	NO	N/A
9	Will your project/assignment deliberately mislead participants in any way?			✓
10	Is there any realistic risk of any participants experiencing either physical or psychological distress or discomfort?			✓
11	Is the nature of the research such that contentious or sensitive issues might be involved?			✓

If you have ticked **Yes** to 9, 10 or 11 you should complete the full Ethics Approval Form. In relation to question 10 this should include details of what you will tell participants to do if they should experience any problems (e.g. who they can contact for help). You may also need to consider risk assessment issues.

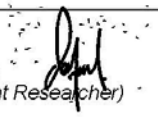
			YES	NO	N/A
12	Does your project/assignment involve work with animals?			✓	
13	Do participants fall into any of the following special groups? **Note that you may also need to obtain satisfactory clearance from the relevant authorities**	Children (under 18 years of age)			✓
		People with communication or learning difficulties			
		Patients			
		People in custody			
		People who could be regarded as vulnerable			
		People engaged in illegal activities (eg drug taking)			
14	Does the project/assignment involve external funding or external collaboration where the funding body or external collaborative partner requires the University to provide evidence that the project/assignment had been subject to ethical scrutiny?			✓	

If you have ticked **Yes** to 12, 13 or 14 you should complete the full Ethics Approval Form. There is an obligation on student and supervisor to bring to the attention of the APU Research Ethics Committee any issues with ethical implications not clearly covered by the above checklist.

STUDENT RESEARCHER

Provide in the boxes below (plus any other appended details) information required in support of your application, THEN SIGN THE FORM.

	Please Tick Boxes
I consider that this project/assignment has no significant ethical implications requiring a full ethics submission to the APU Research Ethics Committee.	✓
Give a brief description of participants and procedure (methods, tests used etc) in up to 150 words. For this research, the dataset will be collected from public domain such as news announcement from financial news website.	
I also confirm that: i) All key documents e.g. consent form, information sheet, questionnaire/interview are appended to this application. Or	✓
ii) Any key documents e.g. consent form, information sheet, questionnaire/interview schedules which need to be finalised following initial investigations will be submitted for approval by the project/assignment supervisor/module lecturer before they are used in primary data collection.	

Signed:  Print Name: Ahmad Fuad Khalit Date: 23st March 2020

(Student Researcher)

Please note that any variation to that contained within this document that in any way affects ethical issues of the stated research requires the appending of new ethical details. New ethical consent may need to be sought.

The completed form (and any attachments) should be submitted for consideration by your Supervisor/Module Lecturer

SUPERVISOR/MODULE LECTURER
PLEASE CONFIRM THE FOLLOWING:

	Please Tick Box
I consider that this project/assignment has no significant ethical implications requiring a full ethics submission to the APU Research Ethics Committee	✓
i) I have checked and approved the key documents required for this proposal (e.g. consent form, information sheet, questionnaire, interview schedule) Or	✓
ii) I have checked and approved draft documents required for this proposal which provide a basis for the preliminary investigations which will inform the main research study. I have informed the student researcher that finalised and additional documents (e.g. consent form, information sheet, questionnaire, interview schedule) must be submitted for approval by me before they are used for primary data collection.	

SUPERVISOR AND SECOND ACADEMIC SIGNATORY

STATEMENT OF ETHICAL APPROVAL (please delete as appropriate)

1) **THIS PROJECT/ASSIGNMENT HAS BEEN CONSIDERED USING AGREED APU/SU PROCEDURES AND IS NOW APPROVED**

2) **THIS PROJECT/ASSIGNMENT HAS BEEN APPROVED IN PRINCIPLE AS INVOLVING NO SIGNIFICANT ETHICAL IMPLICATIONS, BUT FINAL APPROVAL FOR DATA COLLECTION IS SUBJECT TO THE SUBMISSION OF KEY DOCUMENTS FOR APPROVAL BY SUPERVISOR (see Appendix A)**

Signed... ... ... Print Name: Dr. Waddah Waheeb Date: 23rd March 2020
(Supervisor/2nd Marker)

Signed... ... ... Print Name... DR. Raja Rajeswary ... ... Date... 24th March 2020
(Second Academic Signatory)

Office Record	Receipt – Appendix A (Fast-Track Ethics Form)
Date Received: Received by whom:	Student name:Ahmad Fuad Khalit Student number:TP058497 Received by: Date:

APPENDIX A
AUTHORISATION FOR USE OF KEY DOCUMENTS

Completion of Appendix A is required when for good reasons key documents are not available when a fast track application is approved by the supervisor/module lecturer and second academic signatory.

I have now checked and approved all the key documents associated with this proposal e.g. consent form, information sheet, questionnaire, interview schedule

Title of project/assignment: Assignment 2 (Project Proposal)

Improving Stock Market Prediction Performance With Market Sentiment Analysis

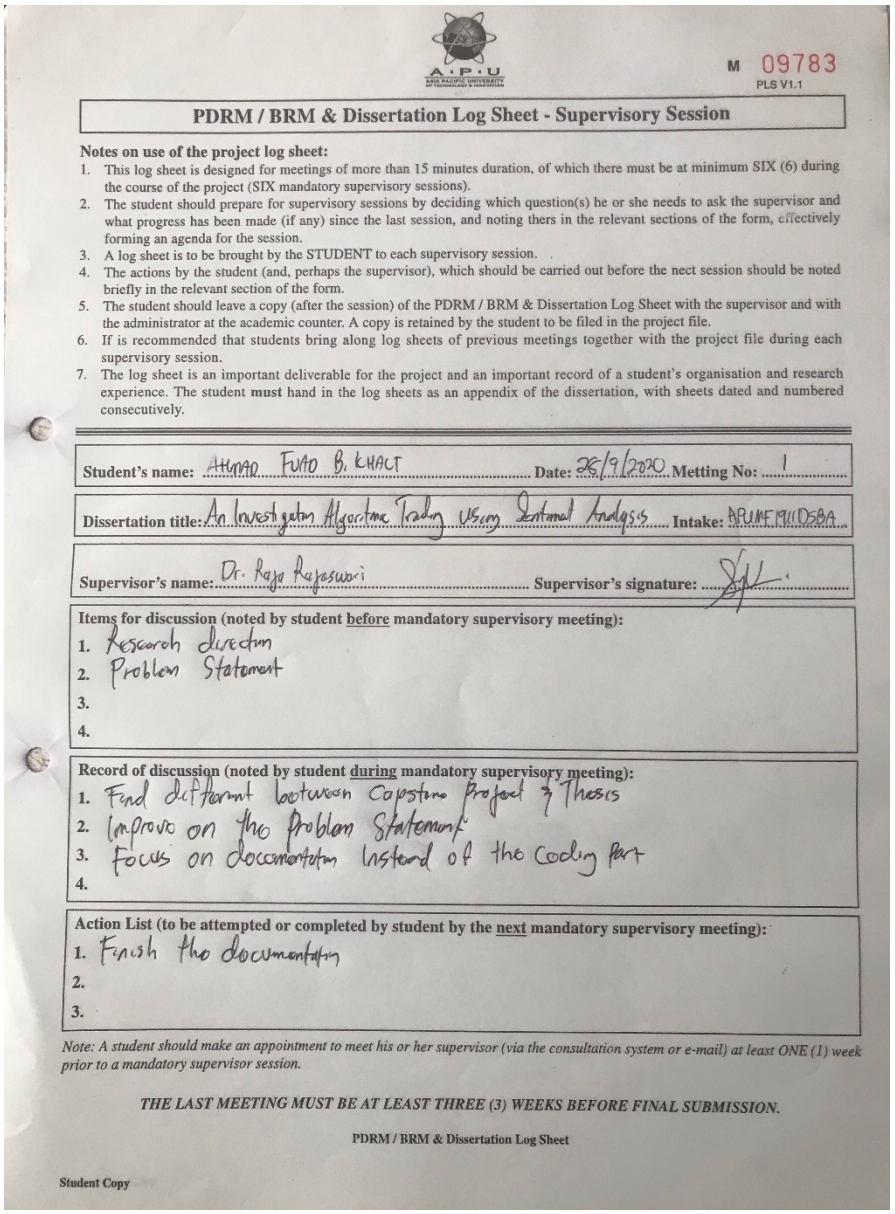
Name of student researcher: Ahmad Fuad Khalit

Student ID: TP058497 Intake: APUMF1911DSBA(PR)

Signed: ..... Print Name Dr. Waddah Waheeb..... Date...23/03/2020 ... .
(Supervisor/2nd Marker)

APPENDIX C

LOG SHEET



A·P·U
ASIA PACIFIC UNIVERSITY OF TECHNOLOGY & INNOVATION

M 09783
PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting thers in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the nect session should be noted briefly in the relevant section of the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. If is recommended that students bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: AHMAD FUAD B. KHALT **Date:** 25/9/2020 **Metting No:** 1

Dissertation title: An Investigation Algorithm Trading Using Sentiment Analysis **Intake:** APUMF1911DSBA

Supervisor's name: Dr. Raja Rajeswari **Supervisor's signature:**

Items for discussion (noted by student before mandatory supervisory meeting):

1. Research direction
2. Problem Statement
3.
4.

Record of discussion (noted by student during mandatory supervisory meeting):

1. Find different between Capstone Project & Thesis
2. Improve on the Problem Statement
3. Focus on documentation instead of the coding part
4.

Action List (to be attempted or completed by student by the next mandatory supervisory meeting):

1. Finish the documentation
2.
3.

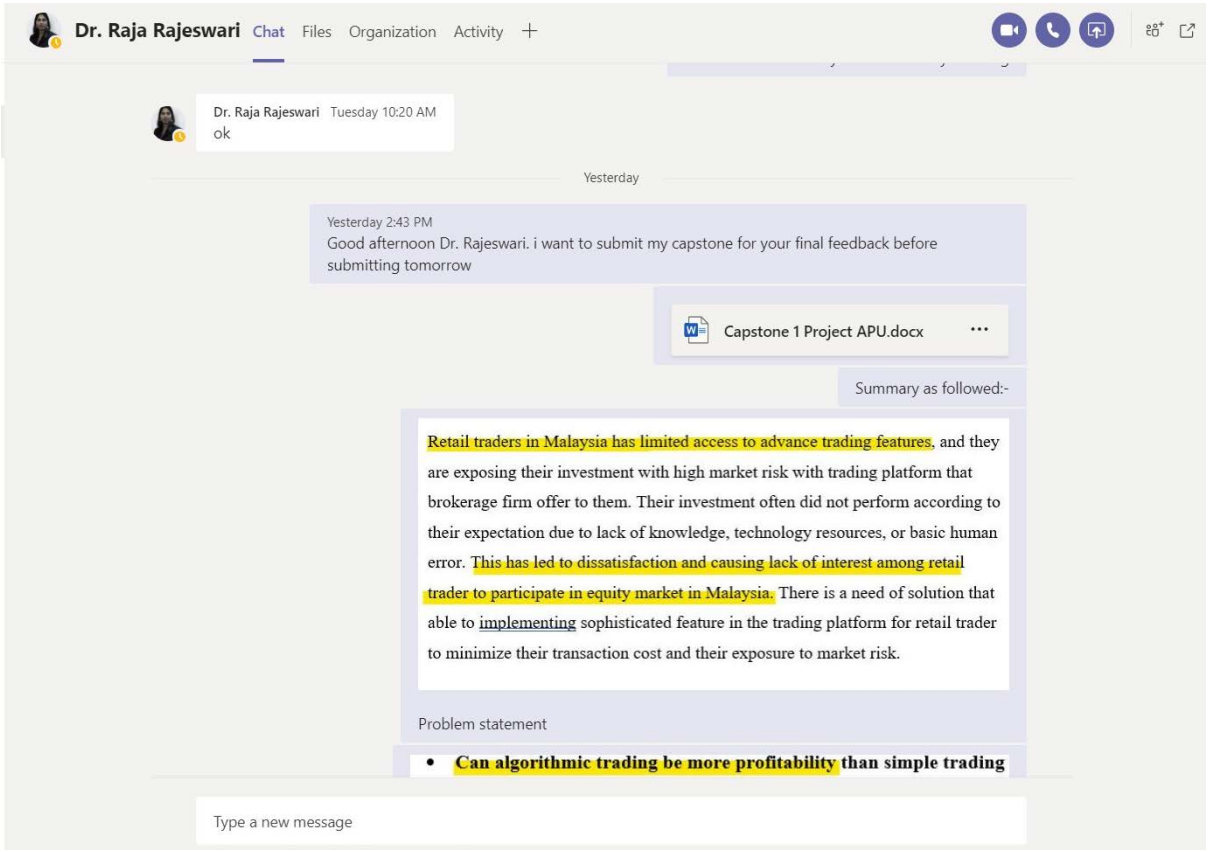
Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

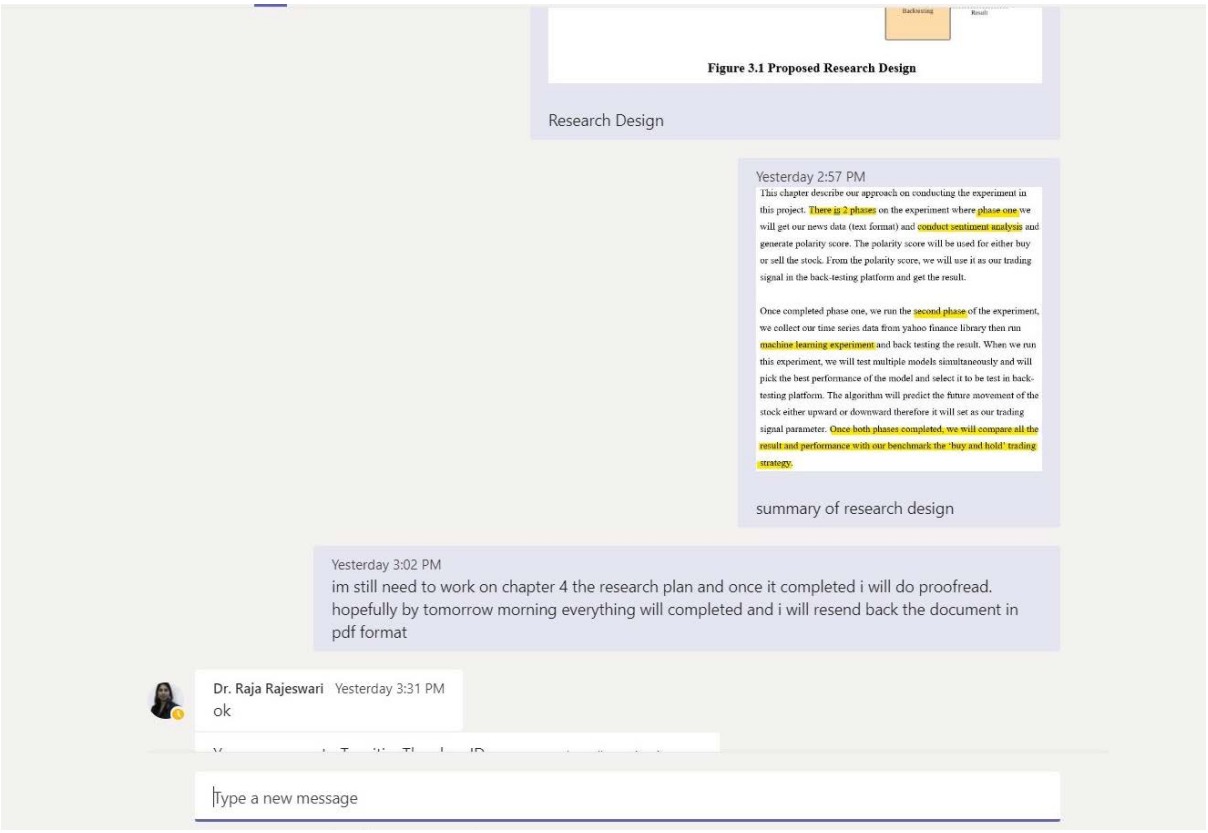
THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.

PDRM / BRM & Dissertation Log Sheet

Student Copy

Online Discussion (MS Team)





A·P·U
ASIA PACIFIC UNIVERSITY OF TECHNOLOGY & INNOVATION

M 09784
PLS V1.1

PDRM / BRM & Dissertation Log Sheet - Supervisory Session

Notes on use of the project log sheet:

1. This log sheet is designed for meetings of more than 15 minutes duration, of which there must be at minimum SIX (6) during the course of the project (SIX mandatory supervisory sessions).
2. The student should prepare for supervisory sessions by deciding which question(s) he or she needs to ask the supervisor and what progress has been made (if any) since the last session, and noting thers in the relevant sections of the form, effectively forming an agenda for the session.
3. A log sheet is to be brought by the STUDENT to each supervisory session.
4. The actions by the student (and, perhaps the supervisor), which should be carried out before the nect session should be noted briefly in the relevant section of the form.
5. The student should leave a copy (after the session) of the PDRM / BRM & Dissertation Log Sheet with the supervisor and with the administrator at the academic counter. A copy is retained by the student to be filed in the project file.
6. If is recommended that students bring along log sheets of previous meetings together with the project file during each supervisory session.
7. The log sheet is an important deliverable for the project and an important record of a student's organisation and research experience. The student **must** hand in the log sheets as an appendix of the dissertation, with sheets dated and numbered consecutively.

Student's name: AHMAD FUAD B. Khalil **Date:** 1/10/20 **Metting No:** 1

Dissertation title: An Investigation on Algorithmic Trading using Sentiment Analysis **Intake:** APUMF1911DSBA

Supervisor's name: Dr. Mohammad Tubishat **Supervisor's signature:**

Items for discussion (noted by student before mandatory supervisory meeting):

1. Introduction
2. Literature Review
3. Research Design
4.

Record of discussion (noted by student during mandatory supervisory meeting):

1. TO improve flowchart of Research Design
2.
3.
4.

Action List (to be attempted or completed by student by the next mandatory supervisory meeting):

1. Re-do flowchart on Research Design
2.
3.

Note: A student should make an appointment to meet his or her supervisor (via the consultation system or e-mail) at least ONE (1) week prior to a mandatory supervisor session.

THE LAST MEETING MUST BE AT LEAST THREE (3) WEEKS BEFORE FINAL SUBMISSION.

PDRM / BRM & Dissertation Log Sheet

Student Copy

January 13, 2021

1/13 10:23 AM
good morning Dr. i have a question. for Simple moving average, if i forecasting 5 days ahead, how to evaluate the performance of forecasting? is MSE/RMSE is enough?

Dr. Raja Rajeswari 1/13 10:25 AM
good morning, yes MSE and RMSE can be used

January 19, 2021

1/19 4:25 PM
Good evening dr. i need some clarification. from the feedback of my CP1 when we use machine learning, the evaluation of the model will always use accuracy, precision, recall etc

but when i look at other researcher work, they normally use sharpe ratio, volatility and other when using machine learning in stock market domain. so i unable to compare it side by side. is it ok if i use the sharpe ratio, volatility etc as well so i can compare it with other work?

this are the sample of previous work i found that identical to my model

Xiong, Z., Liu, X.Y., Zhong, S., Yang, H. and Walid, A., 2018. Practical deep reinforcement learning approach for stock trading. *arXiv preprint arXiv:1811.07522*

Johnman, M., Vanstone, B.J. and Gepp, A., 2018. Predicting FTSE 100 returns and volatility using sentiment analysis. *Accounting & Finance*, *58*, pp.253-274.

Berutich, J.M., López, F., Luna, F. and Quintana, D., 2016. Robust technical trading strategies using GP for algorithmic portfolio selection. *Expert Systems with Applications*, *46*, pp.307-315.

or i should stick with accuracy measurement

Dr. Raja Rajeswari 1/19 7:24 PM
you can use both measurement for comparison

1/19 7:33 PM 👍 1
ok noted. thank you dr.

January 27, 2021

1/27 11:15 AM
Morning Dr Rajeswari. If i made some minor changes from what i wrote in CP1, should i make 1 subtopic just to explain the changes i made or i can just write a sentence to explain it without making specific subtopic for it

in CP1 i wrote about using Vader Lexicon but CP2 i use Loughran-McDonald finance lexicon. it give me more accurate result that why i change

Dr. Raja Rajeswari 1/27 12:11 PM
Hi, need to explain in detail

In methodology

1/27 12:17 PM
noted

APPENDIX D

TURNITIN REPORT

An Investigation on Algorithmic Trading Profitability with Sentiment Analysis Algorithm in Malaysia Market

ORIGINALITY REPORT

9%	%	9%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Anindya Chakrabarty, Anupam De, Rameshwar Dubey. "A Flexible Approach Towards Multi-frequency Re-engineering of the Moving Average Convergence Divergence Indicator", Global Journal of Flexible Systems Management, 2014 Publication	1%
2	"Data Engineering and Communication Technology", Springer Science and Business Media LLC, 2020 Publication	1%
3	Lee Zhong Zhen, Yun-Huoy Choo, Azah Kamilah Muda, Ajith Abraham. "Forecasting FTSE Bursa Malaysia KLCI trend with Hybrid Particle Swarm Optimization and Support Vector Machine technique", 2013 World Congress on Nature and Biologically Inspired Computing, 2013 Publication	1%
	Ming Zhan, Ruibo Tu, Qin Yu. "Understanding	

4	Readers", Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence - CSAI '18, 2018 Publication	1%
5	"Neural Information Processing", Springer Science and Business Media LLC, 2018 Publication	1%
6	"Applied Soft Computing and Communication Networks", Springer Science and Business Media LLC, 2020 Publication	1%
7	Chen Ming-Hsiang. "A Timing Strategy for Investments in U.S. Hospitality Stocks", Journal of Hospitality & Tourism Research, 2010 Publication	1%
8	Jagdish Chakole, Manish Kurhekar. "Trend following deep Q-Learning strategy for stock trading", Expert Systems, 2019 Publication	1%
9	Mazen Nabil Elagamy, Clare Stanier, Bernadette Sharp. "Stock market random forest-text mining system mining critical indicators of stock market movements", 2018 2nd International Conference on Natural Language and Speech Processing (ICNLSP), 2018 Publication	1%
	D. Tamil Priya, J. Divya Udayan. "Transfer	

Technology, 2020
Publication

11 Ayman E. Khedr, S.E.Salama, Nagwa Yaseen. "Predicting Stock Market Behavior using Data Mining Technique and News Sentiment Analysis", International Journal of Intelligent Systems and Applications, 2017
Publication

12 Irfan Kareem, Shahid Mahmood Awan. "Pakistani Media Fake News Classification using Machine Learning Classifiers", 2019 International Conference on Innovative Computing (ICIC), 2019
Publication

13 Marcus Schulmerich, Yves-Michel Leporcher, Ching-Hwa Eu. "Applied Asset and Risk Management", Springer Science and Business Media LLC, 2015
Publication

14 Hui, Eddie C. M., Sheung-Chi Phillip Yam, and Si-Wei Chen. "Shiryaev-Zhou index â€“ a noble approach to benchmarking and analysis of real estate stocks", International Journal of Strategic Property Management, 2012.
Publication

Exclude quotes	Off	Exclude matches	< 1%
Exclude bibliography	Off		