CHAPTER III : RESEARCH METHODOLOGY

3.1 Introduction

Chapter III provides the methodology of forecasting stock prices using six ML models. From the findings in literature review, this research has an aim to evaluate the performance of ML models as a trading strategy in predicting stock movement from traditional ML and deep learning models. This chapter has eight sections, which includes Section 3.1 on the introduction of the chapter, Section 3.2 on the ML models framework in predicting stock price, Section 3.3 is on the data acquisition in the study, and Section 3.4 focuses on data preparation. Followed by Section 3.5 explains on the learning algorithms which includes ML algorithms and Walk-Forward Analysis method, Section 3.6 describes on the performance calculation which includes directional and performance evaluation indicators, Section 3.7 assess on the statistical testing method to test significant difference between all the evaluation indicators used, and lastly Section 3.8 on the summary of all the sections to conclude the chapter.

3.2 Conceptual Framework

The general framework of forecasting the future stock price based on ML algorithms used for this study is demonstrated in Figure 3.2-1 below. This framework is structured from data acquisition to statistical testing. The process starts with the data acquisition which includes data source and software. In data preparation, the stock data is adjusted for ex-dividend/rights, features are generated based of popular technical indicators and normalised so that later it can be used as ML algorithms input (Dongdong et al., 2019). In the next process is the learning algorithm that consists of ML algorithms, Walk-Forward Analysis (WFA) method, and trading signals algorithm design. Followed by performance calculation which includes the directional

evaluation indicators, performance evaluation indicators, and backtesting algorithms. Finally, the models perform statistical testing that consists of Kruskal-Wallis Test and Nemenyi Post-hoc Test.

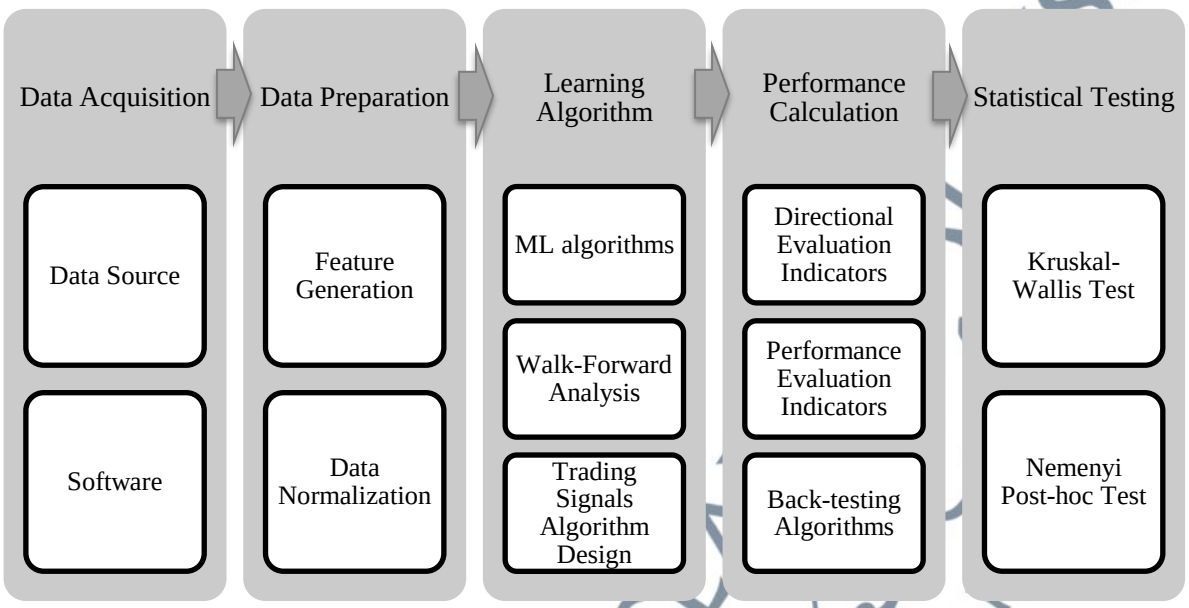

Figure 3.2-1: ML Algorithms Framework in Predicting Stock Price.

3.3 Data Collection

Thirty stocks in Bursa Malaysia were chosen as datasets for this study. The stocks selected are also known as largest thirty companies listed in the Main Board by full market capitalisation that fulfilled the requirements from the Financial Times Stock Exchange Group (FTSE) Bursa Malaysia Ground Rules. Other than the stocks, the index price from Bursa Malaysia KLCI is also downloaded. The data of daily stock is taken from Yahoo! Finance which includes open, close, high, low, adjusted close prices and volume by using *getsymbols()* in *quantmod* package that is used R language in the StudioR software.

3.4 Data Preparation

The features used in this study is listed in the Appendix A in Appendices. It consisted of two components, which are Basic Symbols and Technical Indicators. The selected technical indicators are chosen following on the readings and rationale indicated from past studies. In this study, the number of features or technical indicator, n is set to be twenty for the data of

each trading day. The technical indicators were computed based on the adjusted stock prices and volume by using their equations and normalized. The data is processed and cleaned to give each stock a total of 2,000 trading days of twenty parameters before date February 1st, 2018. All the values are in decimal point between 0 to 1 where it can be interpreted as percentage as 0% to 100%.

3.5 Learning Algorithm

Section 3.5 describes on learning algorithm and it has three subsection, which are ML algorithms, Walk-Forward Analysis (WFA) method, and trading signals algorithm design.

3.5.1 ML Algorithms

Three traditional ML models selected are Logistic Regression (LR), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB). Three Deep Learning models selected are Deep Belief Network (DBN), Multilayer Perception (MLP), and Stacked Auto-Encoder (SAE) to foresee the direction of the stock prices. The ML models are chosen because these models have been studied from the past research and it gives foundation of how good the model in prediction. Another reason is because there is a lack of data analysis made for both traditional ML and deep learning models simultaneously, thus this study aims to evaluate the efficiency of both models.

In Table 3.5-1 and 3.5-2, it shows the setting of main parameters for the ML algorithms used for the study. The input format for all ML algorithm is Matrix (m, n) that denotes a matrix with m rows and n columns. As the input, the data of past trading days, m = 250 is set as training samples with the technical indicators, n = 20. The output in predicting the direction of the stock price also known as the trading signal as shown in Table 3.5-1 and 3.5-2 is called Labels. Next, the dimensions of hidden layers as vector of c(h_1, h_2, h_i) which the vector's length

is the hidden layers number and the *i*-th component of c is the *i*-th hidden layer's neurons number.

Table 3.5-1: Setting of main parameters used for the traditional ML algorithms.

	Input	Label	Parameter
LR	Matrix(250,20)	Matrix(250,1)	Logit model link function is used.
SVM	Matrix(250,20)	Matrix(250,1)	Radial Basis kernel is used.
XGB	Matrix(250,20)	Matrix(250,1)	Maximum depth of tree is 10, iteration maximum number is 15, learning rate is 0.3.

Table 3.5-2: Setting of main parameters used for the Deep Learning algorithms.

	Input	Label	Dimensions of hidden layers	Activation function	Learning rate	Batch size	Epoch
DBN	Matrix(250,20)	Matrix(250,1)	c(25,25,20,5)	sigmoid	0.8	100	3
MLP	Matrix(250,20)	Matrix(250,1)	c(25,25,20,5)	sigmoid	0.8	100	3
SAE	Matrix(250,20)	Matrix(250,1)	c(20,10,5)	sigmoid	0.8	100	3

For traditional ML, it uses different packages to formulate the function for the model. (1) Logistic Regression algorithms' equation can be modelled using *glm()* by setting the argument *family = binomial* in StudioR for Logit model link function. The *glm()* is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution (R Core Team, 2020). (2) Support Vector Machine can be modelled using *ksvm()* in the *kernlab* package by setting the argument *kernel = "rbfdot"* in StudioR (Karatzoglou et. al, 2004). The *rbfdot* is Radial Basis kernel "Gaussian" used when the boundaries are hypothesized to be curve shaped. (3) Extreme Gradient Boosting can be modelled using *xgboost()* in the *xgboost* package by setting the argument *objective="binary:logistic""* in StudioR (Chen, et al., 2020). This will give output of probability when using logistic regression for binary classification.

For Deep Learning algorithms, it uses the same package called *deepnet* package in StudioR (Rong, 2014). The function can either operate regression or classification, depending upon its activation function. (1) Deep Belief Network algorithm can be modelled using

dbn.dnn.train() in the *deepnet* package. Next, (2) Multilayer Perception can be modelled using *nn.train()* in StudioR. Lastly, (3) Stacked Auto-Encoder can be modelled using *sae.dnn.train()* in the *deepnet* package in StudioR.

3.5.2 Walk-Forward Analysis (WFA) Method

Walk-Forward Analysis (WFA) method is a process of optimizing a trading structure using a constrained set of parameters, and then testing out-of-sample data with the best optimized parameter set. Instead of working with the past data, the model which used recent data is trained to carry out the forecast for the out-of-sample data of the future (testing dataset). Then, a new training set is trained for the next round which the previous training set will walk one step forward. This will enhance the robustness and the conviction of the real-time stock trading movement.

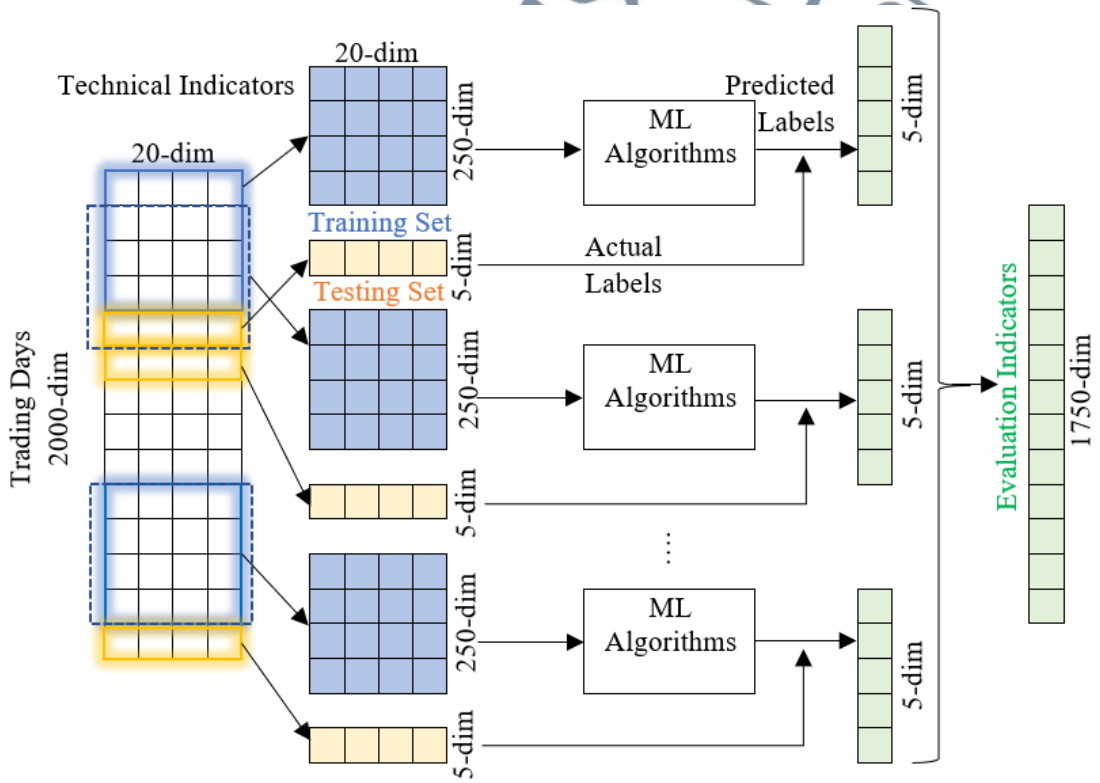

Figure 3.5-1: Schematic diagram of WFA Method.

For this study, the training and the testing set can be seen shown in Figure 3.5-1. For each step, the past 250 trading days (one year) of stock data is used as the training set and the

testing set is set based on the data for the next 5 trading days (one week). The training set will undergo ML algorithms to generate predicted labels. Then, it steps forward by 5 trading days to repeat the steps. For every stock, each datasets includes 2,000 trading days of data, so it takes (2,000-250)/5 = 350 training sessions that will generate 1,750 predictions of trading signals for daily trading strategy.

3.5.3 Trading Signals Algorithm Design

By applying Walk-Forward Analysis method to train each of the ML algorithm, the trading signals is generated to forecast the direction of the stocks. The output of the algorithm is the prediction of trading signal where "1" is the trading signal going up and "0" is the trading signal going down. Figure 3.5-2 shows the algorithms for generating trading signals by using ML algorithms.

```
Input: Stock Symbols
Output: Trading Signals
(1) N=Length of Stock List #30 stocks from BURSA
(2) L=Length of Trading Days #2000 trading days
(3) P=Length of Features #20 technical indicators
(4) k= Length of Training Set for WFA #250 trading days
(5) n= Length of Testing Set for WFA #5 trading days
(6) for (i in 1: N) {
(7)       Stock=Stock Symbols[i]
(8)       M=(L-k)/n
(9)       Trading Signal=NULL
(10)      for (j in 1:M) {
(11)          Dataset= Stock[(k+n*(j-1)):(k+n+n*(j-1)), 1:(P+1)]
(12)          Train=Dataset[1:k,1:(1+P)]
(13)          Test= Dataset[(k+1):(k+n),1:P]
(14)          Model=ML Algorithm(Train)
(15)          Probability=Model(Test)
(16)          if (Probability>=0.5) {
(17)               Trading Signal0=1
(18)          } else {
(19)               Trading Signal0=0
(20)          }
(21)      }
(22)      Trading Signal=c (Trading Signal, Trading Signal0)
(23) }
(24) return (Trading Signal)
```

Figure 3.5-2: Algorithms for generating trading signals by using ML algorithms in R language.

3.6 Performance Calculation

Based on the trading signals, the performance of each model is analysed by using evaluation indicators. It is divided into two parts, which is directional and performance. The directional evaluation indicators are used to evaluate the ML models used in predicting the direction of the stock price in classification and regression while the performance evaluation indicators are used to evaluate the ML models as trading strategies in terms of profitability and risk evaluation. It is worth to mention that the term ML model can be used interchangeably with trading strategy. For directional evaluation indicators, the number of trading strategy is six which consists of six selected ML models. Meanwhile for performance evaluation indicators, the number of trading strategy is eight that consists of six ML models with additional two strategies which are 'Buy and Hold' (B&H) strategy and the benchmark index, KLCI for comparison.

3.6.1 Directional Evaluation indicators

The output of the ML algorithms is an indicator to foretell the direction of the stock. The sequences of sets {UP, DOWN} is set as the actual label of the dataset. For this study, "UP" label is when the price of that day increases from the previous day while "DOWN" label is when the price of that day decreases from the previous day.

Table 3.6-1: The matrix of two categorization results of ML

		Predicted Labels	
		UP	DOWN
Actual Labels	UP	TU	FD
	DOWN	FU	TD

*T = True, F = False

As a result, four classifications of actual and predicted labels are produced as shown in Table 3.6-1, which are TU, TD, FU, and FD. TU is the number of UP that both the actual and predicted labels are UP; TD is the number of DOWN that both the actual and the predicted

labels are DOWN; FU is the number of UP that the predicted labels are UP, but actual is DOWN; FD is the number of DOWN that the predicted label values are DOWN, however the actual is UP.

To evaluate the efficiency of ML models in classification and regression for price forecasting, there are six directional evaluation indicators used in this study. The directional evaluation indicators are Accuracy Rate (AR), Precision Rate (PR), Recall Rate (RR), F1 Score, Area Under the Curve (AUC) and Mean Square Error (MSE).

Accuracy Rate (AR) is defined as the number of predictions that is true to the total number of predictions ratio shown in Equation 3.1 (Dongdong et al., 2019). It is used for categorization and classification.

$$AR = \frac{(TU + TD)}{(TU + FD + FU + TD)} \tag{3.1}$$

Other than that, it is essential to use Precision Rate (PR) and Recall Rate (RR) to evaluate categorization results. In evaluating the significance of outcome, initially these two evaluation indicators are used in the information reclamation field. PR is the number of correctly predicted UP to all predicted UP ration shown in Equation 3.2. High PR value indicates that ML algorithms focuses on “UP” as compared to “DOWN”. RR is the number of correctly predicted “UP” to the actual labelled “UP” ratio shown in Equation 3.3. High RR indicates ML algorithms effectively identify and obtain a large number of “UP”. In reality, it is tricky to produce an algorithm that has a high PR and RR at one time.

$$PR = \frac{TU}{(TU + FU)} \tag{3.2}$$

$$RR = \frac{TU}{(TU + FD)} \tag{3.3}$$

Hence, it is important to calculate the categorization ability of the ML algorithm that has both PR with RR by using a new evaluation indicator. In this case, F1-Score becomes the harmonic average of PR and RR as shown in Equation 3.4. Therefore, F1 is much more

thorough evaluation indicator (Huigol, 2019). When calculating F1, it seems that the weights of PR and RR are equal, but the notion might not always be correct. It is important to note that high F1 score correspond to a good result.

$$F_1 = 2 * \frac{PR * RR}{(PR + RR)} \tag{3.4}$$

Additionally, area under the curve (AUC) is also computed by using *performance* in *ROCR* package. In general, an acceptable AUC value is at 0.5 which suggests no discrimination. More than that, is considered better. In order to compare with past literature, Mean Square Error (MSE) is also used as a directional evaluation indicator in regression of the algorithm. MSE is a risk function, corresponding to the expected value of the squared error loss as shown in equation 3.5. It can be modelled using *MSE()* in StudioR (Yan, 2016). A low MSE indicates a good result.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2 \tag{3.5}$$

where *N* is the number of data, y_i is the actual labels, and $\hat{y}$ is the predicted labels.

With this, there is six result matrices for directional evaluation indicators that can be discussed in data analysis, which is AR, PR, RR, F1 score, AUC and MSE values.

3.6.2 Performance Evaluation indicators

Performance Evaluation Indicators are used to evaluate the stocks in terms of profitability and the risk control ability. For performance evaluation indicators, the number of trading strategy is eight. Aside from the six ML models, the two additional strategies are 'Buy and Hold' (B&H) strategy and the benchmark index, KLCI.

B&H strategy is defined as an investment strategy where the investor buy stocks and holds them for a period of time. B&H is also known as a passive investment strategy. Based on the Labels (output), if the next trading day's signal is "1" or "UP", the stock will be bought

on that day's price and be sold at the next day's price. If "0" or "DOWN", no stock will be trade.

All the trading strategies are used to conduct backtesting and compute the performance evaluation indicators. The backtesting period must be long enough and have a big number of historical data to minimize biasness in sampling data. Therefore, this study uses a big dataset of 1,750 trading signals for each stock for backtesting. Figure 3.6-1 shows the backtesting algorithms for all the evaluation indicators used in the study.

There are four performance evaluation indicators used in this study. The performance indicators are Winning Rate (WR), Annualized Return Rate (ARR), Annualized Sharpe Ratio (ASR) and Maximum Drawdown (MDD).

```
Input: TS #trading signals of a stock
Output: AR, PR, RR, F1, AUC, MSE, WR, ARR, ASR, MDD
(1) N=length of Stock List #30 stocks from BURSA
(2) Bt=Benchmark Index["Closing Price"] #benchmark index closing price
(3) AR=NULL; PR=NULL; RR=NULL; F1=NULL; AUC=NULL; MSE=NULL
(4) WR=NULL; ARR=NULL; ASR=NULL; MDD=NULL
(5) for (i in 1: N) {
(6)       Stock Data=Stock Code List[i]
(7)       Pt=Stock Data ⌊"Closing Price"⌋
(8)       Labelt= Stock Data ["Label"]
(9)       BDRRt=(Bt-Bt-1)/ Bt-1 #benchmark daily return
(10)      DRRt= (Pt -Pt-1)/Pt-1 #B&H strategy daily return rate
(11)      TDRRt=lag (TSt)*DRRt #trading daily return rate
(12)      Table=Confusion Matrix(TS, Label)
(13)      AR[i]=sum(adj(Table))/sum(Table)
(14)      PR[i]=Table[2, 2]/sum(Table[, 2])
(15)      RR[i]=Table[2, 2]/sum(Table[2, ])
(16)      F1=2*PR[i]*RR[i]/(PR[i]+RR[i])
(17)      Pred=prediction (TS, Label)
(18)      AUC⌊i⌋=performance (Pred, measure="auc")@y.values⌊⌊1⌋⌋
(19)      MSE[i]=MSE(TS, Label)
(20)      WR[i]=sum (TDRR>0)/sum(TDRR=/0)
(21)      ARR⌊i⌋=Return.annualized (TDRR)#Used for TDRR, BDRR, or DRR
(22)      ASR[i]=SharpeRatio.annualized (TDRR)#Used for TDRR, BDRR, or DRR
(23)      MDD[i]=maxDrawDown (TDRR)# Used for TDRR, BDRR, or DRR
(24)      AR=c(AR, AR[i])
(25)      PR=c(PR, PR[i])
(26)      RR=c(RR, RR[i])
(27)      F1=c(F1, F1[i])
(28)      AUC=c(AUC, AUC⌊i⌋)
(29)      MSE=c(MSE, MSE[i])
(30)      WR=c(WR, WR[i])
(31)      ARR=c(ARR, ARR[i])
(32)      ASR=c(ASR, ASR[i])
(33)      MDD=c(MDD, MDD[i])
(34)      }
(35)      Performance= cbind(AR, PR, RR, F1, AUC, MSE, WR, ARR, ASR, MDD)
(36) return (Performance)
```

Figure 3.6-1: Algorithms for evaluation indicators of daily trading strategy in R language.

Winning Rate (WR) is the ratio of number of positive earning days to total number of trading days. In other words, it is a measurement of accuracy for the trading signals. The higher the value, the better the performance in profitability of the stock. AR to ML algorithms is similar to WR to trading strategy.

Annualized Return Rate (ARR) is defined as the theoretical rate of return of trading strategy. It is calculated as the return received on an annual basis. Therefore, it is computed based on time-weight. The higher the positive the value, the better the performance of the trading strategy.

Annualized Sharpe Ratio (ASR) is a measure of risk-adjusted return on annual basis. In other words, the ratio of an extra return from an extra volatility endured for holding a risky asset. In this case, the risk-free return or benchmark is set to zero. In general, value greater than one is considered as acceptable. A ratio below than one is considered suboptimal.

Lastly, Maximum Drawdown (MDD) is defined as the biggest decline in the price or value in trading strategy. It also represents the risk assessment in the trading strategy. In general, it is preferred to have a low maximum drawdown as it indicates the small losses in the investment. ARR, ASR and MMD used the same package called *PerformanceAnalytics* in StudioR.

3.7 Statistical Test

For this part, the study focuses on testing the significant difference of various algorithm trading strategy in forecasting used on the outcomes found above. A non-parametric statistical test is applied when the outcome is generated based on no assumption is made on the parameters of a population distribution. It is also used when the data does not follow normal distribution. Therefore, it is understandable that this study uses non-parametric statistical test and not use t-test in analysis of variance.

With that, Kruskal-Wallis Test analysis is applied for this study to test whether evaluation indicators are statistically significant to the study. It can be modelled by using *kruskal.test()* in StudioR. The null hypothesis as shown below stated that the evaluation indicators, *j* of all the strategies are the same where *j* is consisting of AR, PR, RR, F1, AUC, MSE, WR, ARR, ASR, and MDD. There are eight strategies consisted of Index, B&H strategy, LR, SVM, XGB, DBN, MLP, and SAE. Thus, the alternative hypotheses are when the evaluation indicators, *j* of all the strategies are not the same.

$$H_{0,j}: the\ evaluation\ indicator\ j\ of\ all\ strategies\ are\ the\ same$$

$$H_{A,j}: the\ evaluation\ indicator\ j\ of\ all\ strategies\ are\ not\ the\ same$$

Next, Kruskal-Wallis Post-hoc Test is used when overall group mean is shown to be statistically significant. It is also known as Nemenyi Post-hoc Test is used to determine whether the differences for medians of between the trading strategies are statistically significant or not. Then, the p-value is compared to the significance level to assess the hypotheses. This study uses a significance level of 0.05. It can be modelled using *posthoc.kruskal.nemenyi.test()* in the *PMCMR* in StudioR (Pohlert, 2014). Therefore, various comparison analysis between the directional and performance evaluation indicator of any two trading algorithms were assessed and discussed based on the results metric.

3.8 Conclusion

This chapter provides a comprehensive explanation on the methodology of forecasting stock movement using various ML models. This study aims to analyse the performance of ML models as a trading strategy in predicting stock movement from traditional ML and deep learning models. Thus, procedures are described in detail to fulfil the objectives. In evaluating the efficiency of the ML models in prediction, Walk-Forward Analysis (WFA) method is applied in determining the optimal parameters for models to generate trading signals which increase the robustness of the trading strategies. Then, performance calculation is computed in

R language, which includes the directional evaluation indicators, performance evaluation indicators, and backtesting algorithms. Finally, a statistical testing is performed to test the significant difference of evaluation indicators between ML models in price prediction.