Machine Learning Algorithms for Thai Mutual Funds Performance Prediction

Napop Soontrapong 60090019
Nutthapat Phoomara 60090022

Bachelor of Engineering in Software Engineering
Faculty of Enginerring
King Mongkut's Institute of Technology Ladkrabang
Academic Year 2020

Thesis - Academic Year 2020

Bachelor of Engineering in Software Engineering
Faculty of Engineering
King Mongkut's Institute of Technology Ladkrabang

Title: Machine Learning Algorithms for Thai Mutual Funds Performance

Authors:

1. Napop Soontrapong Student ID: 60090019
2. Nutthapat Phoomara Student ID: 60090022

Approved for Submission

.....

(Assistant Professor Dr. Chaiwat Nuthong)
Advisor

Date 10/6/2021

Abstract

Nowadays, Mutual funds is popular tools in the investment. Apart from using for tax reduction, mutual funds can make income with less risk than stock because they have finance advisors to manage the money. The objective of this research are to create the model to predict the mutual funds' s performance in the next 3 month and recommend the most positive return compare to its risk.

Predicting the independent value from the dependent value is in field of machine learning. So we try to performance the model to predict the future performance of mutual funds using machine learning. The foresting NAV is one of the features. so we use ARIMA time series forecasting model to perform this feature information and combine with other financial ratio as the input of the model. We use at least 2 machine learning algorithm to perform the classification on the dataset. The class indicate direction of the performance in term of Net Asset value(NAV). Up class is the class that NAV will be increase in the next 3 month otherwise it is Down class.

In this research, we expects our model can recommend the mutual funds that have the most positive return in the normal situation or the less negative return if all mutual funds have negative return.

Table of contents

List of figures

List of tables

Chapter 1
Introduction

1.1 Problem description

Forecasting the mutual funds performance is the challenging task, since the Financial time series as the mutual funds data is very difficult to predict because most of the data are non-linear and non-stationary with the high heteroscedasticity (Tabachnick and Fidell, 2001). Nevertheless, the plan of the fund manager is to improve investor's profits with the minimum risk. To solve this problem, many researchers try to develop many suitable forecasting models to forecast the fund return. Many of these models are based on the fund's historical performance and using some of the portfolio components such as market capitalization, asset allocation and the value-growth score of the funds to achieve the highest profit as possible.

However, most of the studies use some linear model to solve the problem. For example, the CAPM and the factor model which is a popular tool in finance but if the fund has a complex relationship, these methods might not satisfy in order to capture the relationship. Furthermore, most of the financial models are the regression algorithms that provide the estimated return of the target variable over the time period which is not enough to predict the accurate fund return for the investor.

The goal of this research is to create a Mutual fund's selection model based on the fund performance model of each fund to be a tool that helps the investors to decide which fund is worth to invest their money on. By applying some forecasting models along with the machine learning algorithm and using Thai mutual fund data to train our model Along with its essential component of the Mutual fund to get the more accurately forecasting model that can be functional in the real world scenario.

1.2 Objective

In order to indicate the success of the project, several goals have been set. The following list show the goal that need to be satisfied.

- To construct a forecasting model of NAV return of Thai's mutual fund based on its performance record in the past to use as a component to create Mutual fund's selection model.
- To construct a Mutual fund's selection model to select the Thai's mutual fund that gives the investor the increasing return profit at the three months time period.

1.3 Scope of Work

This research focuses on creating a fund selection model using the machine learning algorithm which trains the model by using the forecasting fund performance (NAV) that results from the prediction of NAV in the next three months in the future start from the present date.

The result of this research would be measured in terms of performance measurement and forecasting accuracy. The training model and testing process will be done only on Thai's mutual fund that mainly invest in the stock exchange of Thailand and founded more than 3 years which gathered the data from SEC (The Securities and Exchange Commission of Thailand).

Chapter 2
Literature review

This chapter provides an overview of the existing related research works. The reviews consist of classifying Mutual Funds based on Relative Performance, a hybrid approach for forecasting the net asset values and predicting close price time series data using ARIMA model and Stock Price Prediction Using the ARIMA Model.

2.1 Classifying mutual Funds based on relative performance

From The finance advisor perspective, How well is the artificial neural network can perform the classification on the mutual funds based on relative performance is the question that many researchers want to answer. Adam and Axel(2017)[1] uses an Artificial Neural Network attempting to classify funds into three groups based on eight fund-specific variables

1. IsIndexFund - Indicating if a fund is replicating an index, or is actively managed
2. Style - Value, blend, or growth, represented by a numerical value: 1, 2 or 3
3. M/B - Ratio based on most recently reported book value
4. P/E - Ratio based on trailing 12 months earrnings
5. AUM - Assets under management, in SEK million
6. MCap - Average market cap, in SEK million
7. Returns 12 m - Trailing 12 month total returns in percentage
8. Returns 36 m- Trailing 36 month total returns in percentage

The Classes that the model try to classify are base on Söderberg & Partners' fund analysis groups funds which are

- Red - Funds with weak performance (lowest 20

- Yellow - Funds with medium performance (20-70
- Green - Funds with strong performance (top 30

The model executed on two different data sets which split into train/test following this proportion

- Data set 1: Classification determined by 3-month future performance
  - Training set: Dec 31st, 2010 - Jun 30th, 2015 (6406 data points, 75%)
  - Testing set: Sep 30th, 2015 - Sep 30th, 2016 (2179 data points, 25%)
- Data set 2: Classification determined by 1-year future performance
  - Training set: Dec 31st, 2010 - Sep 30th, 2014 (5617 data points, 74%)
  - Testing set: Dec 31st, 2014 - Dec 31st, 2015 (1995 data points, 26%)

The Artificial Neural Network that Adam and Axel use was defined as a multilayer perceptron applying a backpropagation learning algorithm. The hidden layer size was set to 100 and the data were scaled using Z-score standardization

The model's result is shown in Confusion Matrix

		Predicted			Total actual
		Green	Yellow	Red	
Actual	Green	124	517	2	643
	Yellow	204	902	2	1108
	Red	145	283	0	428
Total predictions		473	1702	4	2179

Table 2.1: confusion matrix of data set 1

		Predicted			Total actual
		Green	Yellow	Red	
Actual	Green	152	406	28	586
	Yellow	246	722	53	1021
	Red	106	258	24	388
Total predictions		504	1386	105	1995

Table 2.2: confusion matrix of data set 2

The confusion matrix showed that the model does not have sufficient performance to apply to an investment advisor. They realized that many funds had consistent weekly data for the majority of the input parameters, but lacked the same frequency for others. This led to the removal of numerous data points .so be able to improve more consistent data should be recorded and made available for analysis. In addition The reason that model predicts disproportionate instances as yellow is presumably since the majority (50%) of the output parameters in the training data belong to that class. So the training data classifications could be distributed differently. For instance, it could be done evenly (33%/33%/33%), thus giving the algorithm a non-skewed perspective regarding the distribution.

2.2 A hybrid approach for forecasting the net asset values

In 2017, Priyadarshini, Govindarajan, and Sharon combined the advantage of the ARIMA model and the computational power of artificial neural network models to give the time series prediction to The Net Asset Values of five of the Indian Mutual Funds.

Due to the complexity of real problems, any single model is not able to give accurate forecasts. In such cases, hybrid models can be used to improve the forecasting performance In the paper, the researcher did not identify all the variables that they used as input variables. They only showed that used The Net asset values of four of the top ten Indian Mutual Funds, 13 macroeconomic and financial variables were used as input variables. They also applied Principal component analysis which is a variable reduction procedure is used in data processing and dimensionality reduction to the input variables.

The hybrid model consists of the ANN model and the ARIMA model. So They follow the bellow step to construct the model:

1. Firstly, They use ANN to fit the non-linear component to produce the series of forecast $\{E_i, i = 1, 2, 3, ... \}$.

2. Secondly, For the linear component, They used the ARIMA model to produce a series of forecasts of linear components $\{Lt_t\}$. Before using ARIMA, normality tests are conducted on the series.

3. Thirdly, integrating the above two components, the final forecast is obtained as

$$\hat{Y}_i = E_i + L_i \tag{2.1}$$

For the performance measurement, they use standard error estimates and The table below shows the standard error estimate of the ARIMA model, ANN model, and Hybrid model for 5 Indian Mutual funds.

Mutual Funds	ARIMA	ANN	ANN-ARIMA
SAHARA	5.1002	5.4087	3.2081
ICICI	4.3573	3.8451	3.367
ING	4.5328	4.3551	3.578
RELIANCE	5.7394	5.5396	4.9323
HDFC	4.4687	3.7374	1.8734

Table 2.3: performance of a hybrid model

The standard error estimate guarantee that the prediction from the hybrid model is fairly accurate more than the single ARIMA or ANN model. The plot of actual rates versus predicted rates also indicates that the model fits the given data well.

2.3 Predicting close price time series data using ARIMA model

Wadi, Mohammad, Ahmed (2018) created a fit model in forecasting accuracy to predict the closed price of the stock using the ARIMA model. They predicted the closed time series data which was collected from Amman Stock Exchange (ASE) from Jan. 2010 to Jan. 2018.

To create the model, the researcher using MINTAB software is a tool to apply the ARIMA model for closed stock market data to implementing a fitted forecasting model. The daily price index of the Arman Stock Exchange (ASE) in each specific time period has been selected in the form of statistical population. About 2000 observations were accumulated for each variable in the time period that has been mentioned.

S.AL Wadi, Mohammad, Ahmed (2018) found fitted ARIMA models by using RMSE criteria and they defined that the less RMSE point the more fitted ARIMA model. The order p, d, q of the ARIMA model they defined to be between 0 and 2 only since the value can not be negative and the estimation will be pointless if it is more than 2. RMSE score is varied in the range between 4.00 and 5.00 based on their dataset. In conclusion they found that ARIMA(2,1,1) was the fittest model with the RMSE score 4.00.

The result of this research, the best model is ARIMA(2,1,1) with RMSE score of 4.00 and the fluctuation of the dataset is discussed and all outlier values have been detected. With this result, They conclude that it can be useful to guide the investor to select the more profitable decision. The ARIMA model gives a good result when emerging the short term forecasting technique. The only limitation of this model is that they use the ARIMA model with only short term forecasting.

2.4 Stock Price Prediction Using the ARIMA Model

Ayodele A., Aderemi O. (2014) try to predict the stock price of Nokia stock index from covers the period from 25th April 1995 to 25th February, 2011 by using the ARIMA model that were obtained by applying the Box-Jenkins method.

The stock data used in their research are historical daily stock prices obtained from two countries stock exchanged. The data consists of four elements which are open price, low price, high price and close price respectively. The closing price is chosen to represent the price of the index to be predicted. Closing price is chosen because it reflects all the activities of the index on a trading day.

They determined the best ARIMA model applying some criteria to each of the stock indexes. First the relatively low score of Bayesian Information Criterion (Bic). A relatively small standard error of regression. The Q-statistics and correlogram show that there is no significant pattern left in the autocorrelation functions (ACFs) and partial autocorrelation functions (PACFs) of the residuals, it means the residual of the selected model are white noise.

The training process is according to the Box-Jenkin method which is first trying to get the stationary time-series dataset then finding the best optimum order to be fitted in the ARIMA model.

For the result they getting the ARIMA(2,1,0) model as the best accurate model with the BIC score of 5.3927 and the standard error of regression of 3.5808 which are quite accurate for forecasting trend of the next short time period but are not that accurate in term of predicting the exact future stock price.

2.5 Predicting the price of Bitcoin using Machine Learning

Mcnally and crew conducted the experiment that tried to create the ARIMA, RNN, and LSTM machine learning model to predict the future Bitcoin cryptocurrency

price and create a categorized model to classify the trend type by using the data of open, close, high, and low from the CoinBase website. The data has taken from 19 august 2013 to july 2016 to construct the regression model. As a result, Mcnally found that the LSTM model give the most accurate model to classify the data and the RNN model given the lowest Root Mean Square Error score in predicted the price with the accuracy percentage at 52.87% and Root Mean Square Error score (RMSE) at 5.45%.

Chapter 3
Background knowledge

3.1 Mutual Fund

A mutual fund is a type of financial vehicle made up of a pool of money collected from many investors to invest in securities like stocks, bonds, money market instruments, and other assets[6]. Mutual Funds are managed by professional money managers who allocate the fund's assets and attempt to produce capital gains or income for the fund's investors.

3.2 Ratio

Geometric mean return

Geometric mean return is used to calculate average rate per period on investments that are compounded over multiple periods.

$$Geometric\ mean\ return = (1 + r_c)^{\frac{1}{n}} - 1 \tag{3.1}$$

where:

r_c = Cumulative return over the entire period
n = Number of equal subset periods to average the return

Beta

Beta is the measurement of risk or volatility associated with a fund in comparison to the market or a benchmark. In our thesis the benchmark is refer to the index of the stock exchange of Thailand.

$$Beta = \frac{Covariance}{Variance} \tag{3.2}$$

$Covariance$ = Measure of the directional relationship between the returns on two assets. (In this case is the relation between NAV of mutual fund and SET index values)

$Variance$ = Measure of how the market move relative to its mean.

3.3 Machine Learning

Machine learning (ML) is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as ”training data”, in order to make predictions or decisions without being explicitly programmed to do so.[10]

3.3.1 Types of machine learning

Mainly There have 3 types of machine learning algorithms that show below

1. Supervised learning

   This machine learning is presented with some example inputs, based on which the desired outputs are to be formed. The computer is made to learn the general rules of converting inputs to outputs

2. Unsupervised learning

   This machine learning does not provide output to be train, so it has to find its own structure to produce an output. Unsupervised learning involves discovering hidden patterns in data on its own. It involves feature learning, which relates to discovering means toward an end

3. Reinforcement learning

   This machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality

3.3.2 Evaluation

The Evaluation is require to see the performance of Machine learning model. To improve or compare the model in the effective way, we have to select the suitable evaluation method.

Binary classification evaluation

1. **Confusion matrix** is evaluation method that allows visualization of the performance of an supervised algorithm which solve the statistical classification problem also known as Error matrix.

		Actual Values: Positive (1)	Actual Values: Negative (0)
Predicted Values	Positive (1)	TP	FP
Predicted Values	Negative (0)	FN	TN

Table 3.1: Confusion matrix

The matrix consist of 4 component

(a) True Positive (TP): the model predict positive and it is true. The actual value is positive.

(b) True Negative (TF): the model predict negative and it is true.

(c) False Positive (FP): the result of model prediction is negative. but the actual value is Positive which is false.

(d) False Negative (FN): the result of model prediction is positive. but the actual value is negative. It is false and make critical problem in the sensitive prediction.

2. **Accuracy** is the ratio of correctly predicted observations to the total observations. Accuracy is work effectively when the dataset is symmetric where the values of

false positive and false negative is almost same.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.3}$$

where TP TN FP FN are from the confusion matrix

3. **Precision** is the ratio of correctly predicted positive observation to the total predicted positive observations. This ration answer the accuracy of predicted positive. High precision is related to low false positive rate.

$$Precision = \frac{TP}{TP + FP} \tag{3.4}$$

where TP FP are from the confusion matrix

4. **Recall** It is the ratio of correctly predicted positive observations to the all observations in actual value is positive. Therefore recall answer the accuracy of positive predict compare to the actual positive value.

$$Recall = \frac{TP}{TP + FN} \tag{3.5}$$

where TP FN are from the confusion matrix

5. **F-Score** or F1-score is the harmonic mean of precision and recall. So it take false positive and false negative into account. F1-score usually use to comapre the performance between the algorithms.

$$F1\text{-}score = \frac{2 * Recall * Precision}{Recall + Precision} \tag{3.6}$$

where Recall and Precision are defined above.

3.4 Time series Data

Time series data is the various sets of data that are collected repeatedly over the time. Usually the time period is made at evenly spaced times like daily, monthly, and yearly.

3.4.1 Time Series Graph

A time series graph takes the observed data to plot a graph that the y-axis is against the increment of time according to the x-axis. These graphs are often used for visualization of data such as their behavior and pattern through time period to build some reliable model.

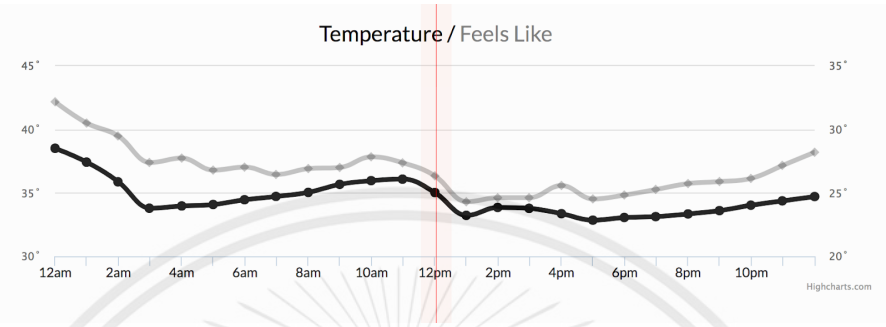

Figure 3.1: A simple example of time series graph

3.4.2 Stationary time series

A Stationary time series occurs when all statistical characteristics of that series such as mean, variance, autocorrelation, etc. are unchanged by shifts in time. Or we can say that all the variables remain constant over time.

Most of the forecasting methods were based on the assumption that the time series can be changed to stationary by applying some method of mathematical transformation. A stationarize series is easy to forecast the future data behavior because we can assume that its statistical properties will be the same as it had been in the past according to the definition of stationary time series.

3.4.3 Differencing

Differencing is a method that transforms the non-stationary time series into stationary. This is also an important step to preparing the data before using it in any time series forecasting model, especially the ARIMA model.

The first differencing value is the difference between the value of data at the current time period and the previous time period. If the different values fail to create some constant statistical properties then the process moves to find the second differencing by using the first differencing value to calculate. Repeated this process until the time series is stationary.

After getting the different values from the time series, take all of the values to plot the differenced series graph to see if there are constant statistical properties and it's sufficient to be the stationary time series.

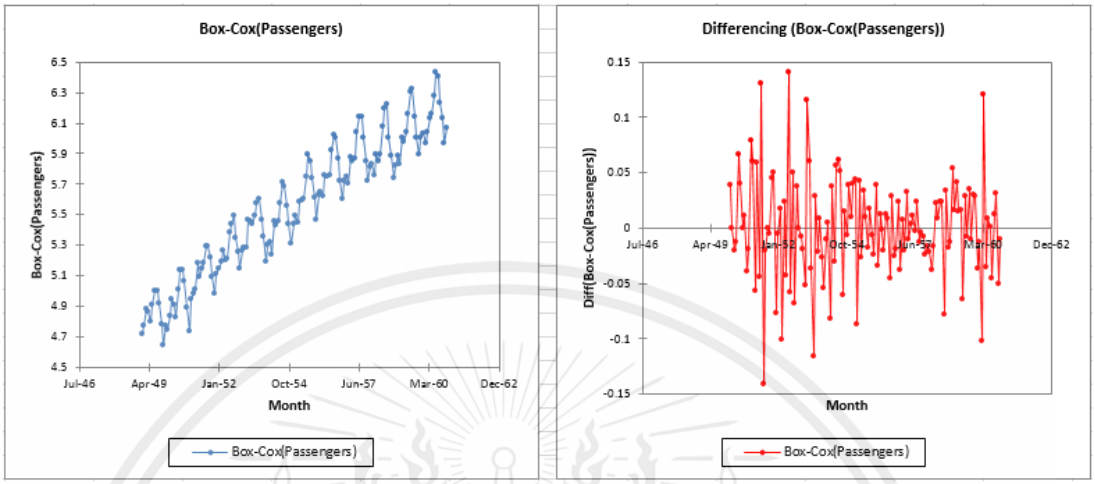

Figure 3.2: An example of time series that had been applied with Differencing method.

3.4.4 White Noise

For the time series to be white noise, it must reach all of the essential conditions which are:

- All of the elements in time series must be independent and identically distributed with the mean of zero.
- The standard deviation of the time series must be constant though time period.
- The correlation between lags is also zero.

White noise time series can not be predictable because by the definition, all of the data in series is random. In order to predict time series data is through the combination of the signal generated by some prediction process with the noise which are the series of errors which we clearly know that it is a white noise time series.

3.4.5 Augmented Dickey-Fuller Test

Augmented Dickey Fuller test is one type of statistical test in the unit root test which are the thing that determine how strongly a time series is define by seasonality

and trend that cause the time series to be non-stationary

The null hypothesis of the test is that the time series can be represented by a unit root, that it is not stationary (has some time-dependent structure). The alternate hypothesis (rejecting the null hypothesis) is that the time series is stationary.

- **Null Hypothesis (H0):** If failed to be rejected, it suggests the time series has a unit root which is non-stationary.
- **Alternate Hypothesis (H1):** The null hypothesis is rejected, it means that time series does not have a unit root, meaning it is stationary.

A p-value in the test is a threshold (10%, 5%, 1%) which normally are 0.05 it is the rate to suggests to reject the null hypothesis witch mean its stationary if it within the threshold range, otherwise a p-value above the threshold suggests that it fail to reject the null hypothesis to be stationary.

3.5 Time Series Forecasting

3.5.1 ACF - Auto-Correlation function

The term correlation refers to how correlated the target time series have according to its past values. Whereas the ACF is the function that calculates the value of auto-correlation of any series and the lagged values of itself. Using this value to plot a graph of ACF plot to describe value quality if the present that is related with the past value. Normally the time series has many components such as its trend and seasonality. ACF is a complete auto-correlation plot which will find the correlation along with all of the components of that time series.
The correlation between two variables y_1, y_2 is defined as:

$$\varrho = \frac{E[(y_1 - \mu_1)(y_2 - \mu_2)]}{\sigma_1 \sigma_2} = \frac{Cov(y_1, y_2)}{\sigma_1 \sigma_2} \tag{3.7}$$

Where E is the expectation operator μ_1, μ_2 are the means for y_1, y_2 and $\sigma_1 \sigma_2$ are their standard deviations

In context of a single variable like auto-correlation, y_1 is the original series and y_2 will be the lagged version of it. Therefore the sample of auto- correlation of k = 0, 1, 2, … can be obtained by calculate using the expression below with the series $y_t, t = 1, 2, ..., n$:

$$p(k) = \frac{\frac{1}{n-k}\sum_{t=k+1}^{n}(y_t - \bar{y})(y_{t-k} - \bar{y})}{\sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \bar{y})^2}\sqrt{\frac{1}{n-k}\sum_{t=k+1}^{n}(y_{t-k} - \bar{y})^2}} \tag{3.8}$$

Where $\bar{y}$ is the sample mean of data

3.5.2 PACF - Partial auto-correlation function

Partial auto-correlation function is not the same as ACF that finds the correlation of present value with lag. PACF finds correlation of the residuals obtained by removing the effects of the earlier lags to correlation with the next lag value. The term "Partial" refers that the already found variation had been removed before finding the next correlation point. So if there is some hidden information in the residual which can be modeled by the next lag, we can assume that the correlation result will be good and the next lag will be used as a feature while doing the modeling. In the modelling process, if there are too many features that are correlated, it can lead to multicollinearity issue problems. Hence when creating the modeling, it should retain only the relevant features.

Partial auto-correlation function measures the linear dependence of one variable after removing the effect of other variables that affect both of it. For example we have ytdata so there are two data that have the effect on y_t which are y_{t-1} and y_{t-2}. In order to measures the partial correlation of y_{t-2} that have some effect on y_t, we have to remove the effect of y_{t-1} on y_t and y_{t-2} first.
Partial auto-correlation can be calculated from the series form of regression as found below:

$$\tilde{y}_t = \phi_1\tilde{y}_{t-1} + \phi_2\tilde{y}_{t-2} + e_t \tag{3.9}$$

where $\tilde{y}$ is the result of original series y_t minus sample mean $\bar{y}$.the estimate value of ϕ_2 given value of partial auto-correlation order 2. Extending the regression with some k additional lags, then the estimate of the last term will give the partial auto-correlation of order k.

3.5.3 Auto regressive (AR) model

For any time series to be an AR, the data that are used to plot time series must be obtained by the previous value or behavior of the same time series. To be precise, we can assume that the present value comes from weighing the average of the past value of itself. As we are going to forecast the NAV return which the data takes some effect from the past value, the AR model is the good one to be using.

The AR(p) model can be written as:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \varepsilon_t \tag{3.10}$$

The p value refer to the order, for example, AR(1) means the first order of AR model which is the AR model that take the data from the first time period and for the second and third order the time period are also different .Where ε_t white noise and the two variables y_{t-1} and y_{t-2} are the lags and the last value C which are constant. Order p is the lag value that PACF plot crosses upper confidence interval at the first time. For forecasting the time series of AR models these p lags will become the feature of the model.

Auto-regressive models are very flexible in order to handle sets of time series that are wide range and have many different patterns. Changing $\phi_1, ..., \phi_p$ will gave us a different time series pattern and variance of ε_t only change the scale of the time series.

3.5.4 Moving Average (MA) model

A moving average model is another type of time series that the present value is obtained as a linear combination of the past errors. So instead of forecasting the present value from the effect of the past value in the form of regression, the MA model uses the previous forecast error in the regression form to forecast present value. To use the MA model, the time series of error needs to be independently distributed with the normal distribution.
The MA(q) model can be written as:

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \varepsilon_{t-q} \tag{3.11}$$

The value q refer to the order as the same as p value in AR model and ε_{t-1} is the error term from the last time period where θ_q is the lag. By changing the lag parameter, its will result in different time series pattern but the noise $\varepsilon_t$are the same as the AR model which will only affect on the scale of the time series only.

Order q of the MA process can be acquired by the lag of ACF that crossing the upper confidence interval for the first time and since the MA time series is the linear combination of residual and its own lag can't define its own present value because it is not AR process so the PACF is no use in this model. MA models also don't have any seasonal or the trending component so the ACF can act as the complete plot method for capturing the correlation due to the residual component only by acting as a partial plot

that works well with the MA model.

3.5.5 Auto Regressive Moving Average (ARMA) model

Auto Regressive Moving Average models are the combination of AR(p) and MA(q) models which are used to forecast the future value in the time series. Where the AR(p) past refers to the regressing process of its own lag with the past value and MA(q) is the linear combination of its past error that is independently distributed with the normal distribution. Merging two of these things together we get the ARMA(p,q) model which is the relationship between the random noise (MA) and its previous value that affect the present(AR) in the time series.
The ARMA(p,q) model can be defined as :

$$y_t = c + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + ... + \theta_q\varepsilon_{t-q} + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} \quad (3.12)$$

The term ε_{t-q} and $y_{t-p}$are refer to the same definition in both AR and MA models.ε_{t-q} is the error term that taken from the past period and y_{t-p} refer to the value in the previous time period. y_t is the current value at the current time period and ε_t are the white noise. The ε_q and ϕ_p re the coefficient with the value associated with the q and p lags respectively.

3.5.6 Auto-Regressive Integrated Moving Average (ARIMA) model

ARIMA model is another subset of linear regression model that combine the previous model which are AR and MR model along with I which are integrated that is denoted as the differencing method to remove some trend in time series to make it more stationary for forecasting. In general the differencing are much more stationary than the raw one. To clarify, the result time series of the ARIMA model should be mean variance stationary so statistical properties of the model is not vary when taken from different time periods.
In the term of mathematical, the ARIMA model now require 3 significant parameter to write in term of ARIMA(p,d,q) which are:

- p → the order of the auto-regressive model.
- d → the degree of differencing.

- q → the order of moving average models.

And the equation can be written as:

$$y'_t = c + \phi_1 y'_{t-1} + ... + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + ... + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{3.13}$$

Where y'_t is the difference series which can be difference more than once and the predictor on the right are the lagged value of y_t , the lagged error and also c and ε_t which are constant and the white noise respectively.

3.5.7 Akaike's Information Criterion (AIC)

The Akaike's Information Criterion is one of a useful method that choose the optimum order (p,d,q) of the ARIMA model

$$AIC = -2log(L) + 2k \tag{3.14}$$

Where L is likelihood of the data and k is the number of parameters. In order to select the best model, we have to select the model with lowest AIC point compared to the point with the other model.

These models can only be used for selecting p,q value but not d which is the order of differencing because the differencing changes the data so the likelihood will be changed which makes the model not comparable with each other.

3.6 Learning Algorithm

3.6.1 Feedforward Neural Network Algorithm

A Feedforward Neural Network is the most basic type of neural network. The information only flows in one direction from the input layer towards the output layer. There is no cycle in the layer and the information does not pass through a single node twice. The network can be represented by a directed graph as shown in Figure 3.3 where nodes represent the neurons and arrows represent the link between them.

Each node in a particular layer is connected to all nodes in the network,it is called a fully connected layer. The connection between node i and node j is called weight coefficient w_ij as shown in the Figure 3.4. The weight coefficient represent the importance of the given connection in the network

Feedforward network can be divided into two passes, the forward pass and the backward pass. The forward pass is where the neural network takes inputs into calcu-

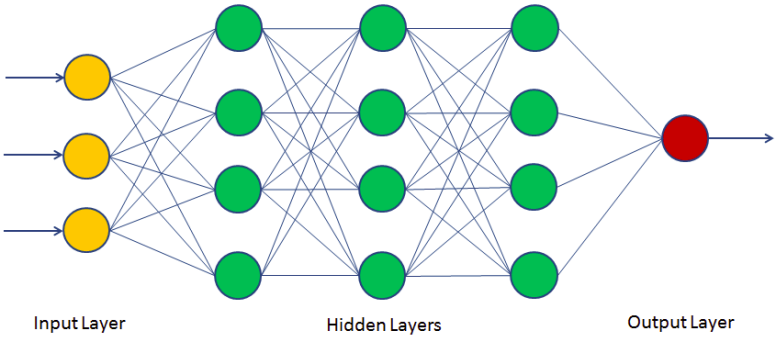

Figure 3.3: A directed graph of neural network with a single hidden layer.

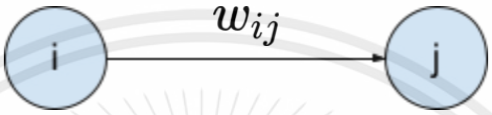

Figure 3.4: Connection between node i and j

lation and traverses through the entire network resulting in a prediction.The backward pass is where the neural network updates its weights to improve its prediction. Normally the newly updated weights are calculated using gradient descent algorithm.The weight is updated from the last layer and traverse backward through the network

The weight will be adjusted along the path in the network until the minimum cost is found during the learning phase. Modifying connection weight can be described by back-propagation network algorithm .The classification phase is used in fixing the amount of weight.

3.6.2 Long Short-Term Memory Network Algorithm(LSTM)

Long Short-Term Memory network(LSTM) is an improved version of a general Recurrent Neural Network (RNN). It's designed to avoid the long-term dependency problem. Learning information over a long period in RNN can cause the information to be lost. The problem is called the vanishing gradient problem. The weight in the recurrent layer will be lower than intended and cause the network to stop learning. LSTM alleviate this problem by adding the ability to select which information to keep or discard through the recurrent step.

Figure 3.5 shows the architecture of a single LSTM cell. The information from the input $(C_t - 1)$ flows towards the output (C_t) of the entire information chain. The sigmoid gate is what chooses which information to let through. The sigmoid function rescaled the value of $h_t - 1$ to be in the range of 0 and 1, denoted as f_t. The rescalling

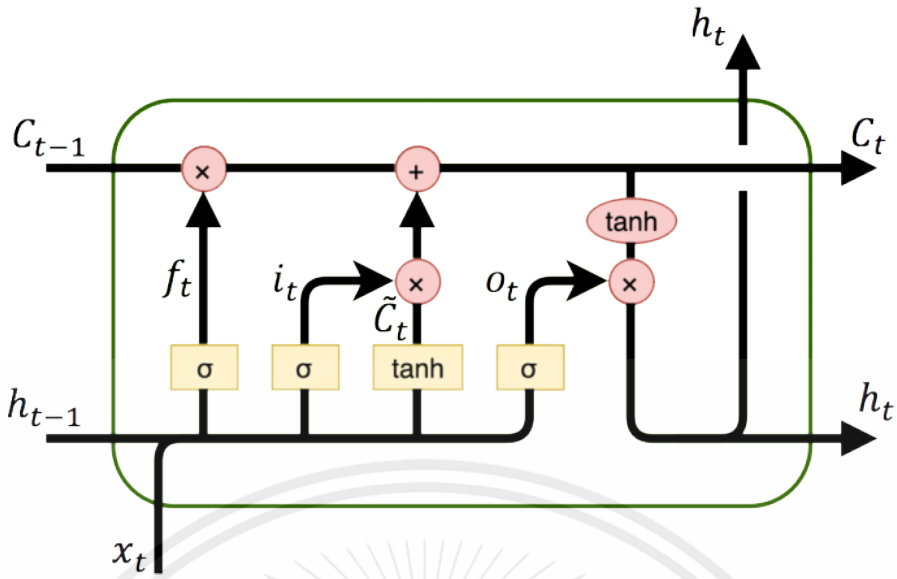

Figure 3.5: A cell of LSTM neural network.

value is then multiplied with the information chain$(C_t - 1)$. By f_t being 0 means that the information is discarded while being 1 result in remembering everything.

The ability of LSTM is to remember new information. This is done using the sigmoid gat on $h_t - 1$ to choose what information to remember and result in i_t. New information is generated by passing $h_t - 1$ to the tanh function that output $\tilde{C}$. By multiplying $\tilde{C}$ and i_t results in the value that is ready to be updated to the information chain. The value is then added to the information chain and result in C_t.

The output of the LSTM cell in Figure 3.5 denotes as h_t. Again the decision of what to output is generated by the sigmoid layer (o_t). The information(C_t) is passed through the tanh function and multiplied with (o_t) resulting in the output (h_t).

LSTM have the cell state to add or remove information, located at the topmost line in the cell according to Figure 3.5. The cell state is connected to the gates in the cell. The gates in the cell are a way to screen the information passing through the cell. The sigmoid layer outputs numbers between 0 and 1 indicating the amount of information to be going through the gate with value one means let all information going forward and vice versa to control the cell state.

3.6.3 Hyper-parameters

Hyper-parameters are parameters that have to be set beforehand manually. it is not learned from the model.

Loss Function

Loss function tells the neural network how good the predictions are. The model will try to optimize the loss function to reduce the errors

Root Mean Square Error

Root mean squared error is of the standard method to measure the error of a model in forecasting the time series data. Root mean squared error is the standard deviation of residuals which are the measurement tool to indicate range between the regression data line and the predicted one. So Root mean squared error can measure how spread out these residuals are to identify the accuracy of the model to be fitted well for the dataset

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y_i} - y_i)^2}{n}} \tag{3.15}$$

3.6.4 Activation Function

Activation function is a thing that squash output from the neuron into a certain range. The function is often non-linear which increases the learning rate of the neural network. Without the non linear activation function the model will just act like a regression model

Sigmoid Function

Sigmoid function rescaled the value to be between 0 and 1. The function can be used in the output of the neural network to give out predictions of classification problems.

$$S(x) = \frac{1}{1 + e^{-x}} \tag{3.16}$$

Swish Function

Swish function is developed to solve the problem of ReLU function. when the negative value pass through to ReLU function the derivative will become 0. It affect to the gradient decent process. The gradient decent not update the solution to become optimal solution.

$$\begin{aligned} f(x) &= x * S(x) \\ &= \frac{x}{1 + e^{-x}} \end{aligned} \tag{3.17}$$

Chapter 4
Methodology

In this chapter, the methodology that used to achieve goals will be discussed. From the beginning point which are collecting data and selecting feature to the process of training and evaluating the dataset.

4.1 Data collecting

The daily NAV of mutual funds are collecting from the SEC Application programming interface (SEC API) provided by The Office of the Securities and Exchange Commission(SEC). The dataset that we create from the gathering API consist of 4 columns.

1. value: The NAV value from the foundation date
2. amount: total assets of mutual fund in Bath
3. updated date: The date for NAV value
4. fund code: The code for mutual fund that register to SEC.

4.2 Feature determination

After gathering the data, We have to consider the features that we will use as the inputs of machine learning model. For now we have 5 features as follow

1. NAV: the daily NAV of mutual fund
2. amount: current amount of the funds
3. return: The return of NAV from previous day.
4. return-3: The geometric mean return of the mutual funds NAV from the previous 3 month to the current date for each daily NAV value.

5. return-6: The geometric mean return of the mutual funds NAV from the previous 6 month to the current date for each daily NAV value.

6. beta-3: The beta ratio of the mutual funds which compute in 3 month scope.

7. SET index The index of the stock exchange of Thailand which correspond to each date of NAV in the dataset.

All of the features can be computed directly by applying some mathematical functions that were commonly used in mutual funds calculation..

4.2.1 Box-Jenkins method

The Box-Jenkins Analysis refers to a systematic method of identifying, fitting, checking, and applying ARIMA models to find the best fit of a time series dataset then applying this optimum ARIMA model to forcasting the NAV return of the next three months. The model indicates 3 steps which are:

1. Model identification
2. Parameter estimation
3. Diagnostic checking

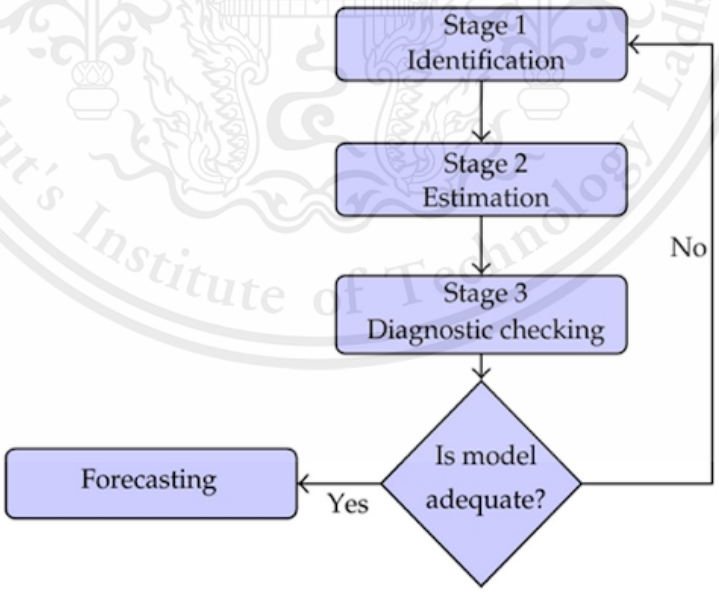

Figure 4.1: The flow of the Box-Jenkins method.

Model Identification

To identify the model, the process was further broken down into two more stages. The first one is to define whether the time series is stationary by using the Augmented Dickey Fuller testing, if not, we do the transformation process which is differencing to create the stationary time-series.

The second stage is to identify the order p,q of AR and MA model respectively. After getting the stationary time series, we plotted the Auto-correlation Function (ACF) and Partial Auto-correlation Function (PACF). From the Graph we can observe the pattern which are:

- If the ACF trails off after a lag and has a hard cut-off in the PACF after a lag. Then we considered to using the AR model only and PACF lag is taken as value for p.
- If the PACF trails off after a lag and has a hard cut-off in the ACF after the lag. Then we considered to using the MA model only and ACF lag is taken as value for q.
- If both ACF and PACF trail off after a lag , then we considered to using both AR and MA model, ACF lag and PACF lag is taken as value for p,q

Parameter Estimation

This step tries to find the best coefficient (pd,q) that will fit well with the ARIMA model. The most common use method is doing the maximum likelihood estimation and checking the Akaike's Information Criterion (AIC).

Diagnostic Checking

Once the model has been fit, the last step is to checking the residual. First if plot the residual graph, it should be close to be white noise as much as possible. Another way is to check the auto-correlation and partial auto-correlation of the residuals. If the values are small, the model is considered adequate and a forecast can be generated. If some of the auto-correlations are large, the values of p and/or q are adjusted and the model is re-estimated.

4.3 Machine Learning Model Construction

4.3.1 Traditional Neural Network Classification with non-time series dataset

We will use mutual fund's NAV combine with other finance ratios as the features to construct machine learning model. We will using the machine learning algorithm to classify the mutual fund into two class.

1. Buy the class indicate mutual fund will have positive return in 3 months
2. Not-buy the class indicate mutual fund will not have positive return after 3 months

Criteria to define class

To define class for each row of mutual funds information in the dataset. we compute the geometric mean return of 3 future month NAV which is out-of-sample. If the geometric mean return is positive then we labeled the target to Buy. otherwise we labeled the target to Not-buy.

4.3.2 Time Series Forecasting Using Long Short-Term Memory Neural Network

Univariate LSTM

The Univariate LSTM focuses on a single dependent variable. The basic assumption behind the univariate prediction approach is the fact that the target value of the time-series data in the current time step is closely related to the target value in the previous time step and as longer the time step is the relation between current target value is slowly decreasing

Univariate model one of the ideal models to use to forecast the future NAV based on the record of NAV in the past. A forecasting model that is trained solely on the basis of NAV development attempts

1. **One-to-one LSTM**

   This is the classic feed forward neural network architecture, with one input the data being fed sequentially and apply to a hidden layer of LSTM units and we expect one output. So the whole sequence of NAV has been grouped together to feed in as a single input and use it to predict the future value one at a time

Many-to-one LSTM

The sequential data is input to the model in each time-step. In this case the time-step is the record of NAV in the past, inputting each value of different time-step into the model. The output vector from the LSTM layer is the last state of the calculated final time-step.

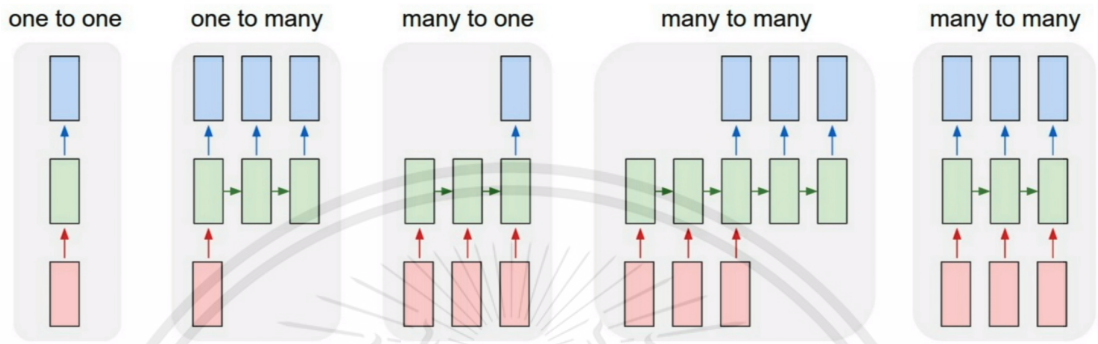

Figure 4.2: Combination of LSTM inputs and outputs.

4.3.3 Hybrid Model

Hybrid model is the combination of the traditional and LSTM network. By using the result of LSTM model as the feature of traditional model.

Architecture design

The result of LSTM model is the future NAV in specific range from the last date in the training data. So we predict next 90 days NAV and apply geometric mean return to the predicted NAV. we will got pred+3 feature, Then we add this feature to the dataset that we define in classification model in section 4.3.1

We will use same configuration as the classification model to compare the performance between the models for describing the benefit of combining 2 different network in this problem.

Chapter 5
Experimentation

5.1 Dataset

the following experiment is done on the dataset that was created from data that were acquired form SEC API (application programming interface of the Office of the Securities and Exchange Commission)

There are a total 156 dataset and each of the dataset consist of 5 rows. The sample dataset is shown in the Figure 5.1 where the “value” column is the value of the daily NAV according to the “updated” column which is the updated date. The next “amount” column is the unit of fund according to the current price and the last “fund-code” column is the identity code of each fund.

Each dataset contain information from the founding date of the fund to 2020-04-20. But the last date that er can use for test is 2021-01-20 (target is valid in this date) and We select 2020-07-08 to be the current date point

5.2 Development Tools

the system is implemented in python and libraries that are used in development are shown in table

Table 5.1: Library used

Libraries	Purpose and Usage
Sklearn	Machine Learning implementation
Pandas	Manage data
Numpy	Computation and Data manipulation
Matplotlib	Data visualisation
Statsmodels	ARIMA Model training
Pmdarima	ARIMA model optimisation
Keras	LSTM network implementation

NAV-SI-1AM-TG

value	amount	updated	fund_code
9.9999	124999162	2008-11-27	1AM-TG
10.0006	125007688	2008-11-28	1AM-TG
10.0053	125067278	2008-12-01	1AM-TG
10.006	127379725	2008-12-02	1AM-TG
10.0067	128026593	2008-12-03	1AM-TG
10.0069	151130373	2008-12-04	1AM-TG
10.0099	151400210	2008-12-08	1AM-TG

Figure 5.1: sample of the dataset.

5.3 Experiment

5.3.1 Predict future performance using classification network

Gathering features information

we add the features which mention in the section 4.2 to the dataset by use the date in the row as the current date for computing feature.

For example,the current date is 2020-07-08. so for feature return-3. the correspond date is 2020-04-08 if the date is validate otherwise it will be the nearest date that not more than 5 dates (For example, in this case is 2020-04-08 to 2020-03-31). then we apply the geometric mean return to compute return of the NAV in this range. This method also apply to the feature return-6 and beta-3 (each NAV value along with the SET index value that have the same date). After add all necessary features, we select every 3 months in the nearest date from the current date back to 2017(3 years) because use all date in the dataset is make model complex and more sensitive to the change of data that cause overfit problem.

	nav	set_index	return-3	return-6	beta-3
0	10.0621	1673.20	0.098419	0.005192	0.002825
1	10.0676	1675.00	0.099506	0.005601	0.003164
2	10.0774	1674.10	0.099471	0.008391	0.003017
3	10.0801	1673.48	0.077670	0.010678	0.003977
4	10.0768	1671.31	0.061664	0.030342	0.004198

Figure 5.2: sample of the dataset with additional features.

Preprocessing data

This the process of preparing data before we apply the data to the model. This process consist of 2 parts following below

1. Splitting train and test set
2. **Scaling data**

   Scikit-learning is provide the Standardscaler class to transform the value of each feature to scaled value. Scaling data reduce the variance effect of the initial variables. Large differences between the ranges of initial variables, those variables with larger ranges will dominate over those with small range.

	nav	set_index	return-3	return-6	beta-3
0	-0.349913	1.121289	0.672687	-0.067995	1.119202
1	-0.349146	1.125754	0.681006	-0.063800	1.316301
2	-0.347780	1.123522	0.680734	-0.035168	1.230939
3	-0.347404	1.121983	0.513974	-0.011698	1.789473
4	-0.347864	1.116600	0.391539	0.190090	1.918321

Figure 5.3: sample of the dataset after scaling.

Implementing models

For the implementation models, we use Keras Sequential model to construct the network. so we try to construct the model with different setting:

number of hidden layers	number of nodes	activate functions
1,2,3	1,5,10,15,...,100	Relu, Sigmoid, Swish

Table 5.2: setting parameter for the neural network

the base compile setting is

```
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

Applying Regularization

In the case that model perform well on the train dataset and has potential to drop its performance in test dataset, we will apply the regularization method which penalty the model when it make false classification to reduce the overfitting problem. Keras Sequential model also support regularization by passing the parameter to model when it construct or fit.

Regularization	Implementation
Early stopping	es parameter of model.fit
l1	kernel_regularizer when add layer to model
l2	activity_regularizer paramerter when add layer to model
dropout	add Dropout layer after each Dense layer

Table 5.3: setting parameter for the neural network

Model Testing

Usually the model is train from the test set which split from the dataset. In our experimentation, we use 70 percent of data to train and 30 percent for test. Testing are perform using Keras model evaluate function, sklearn cofusion matrix and classification_report.we test the model in 3 different way.

1. traditional test

   Testing model as traditional way. Test on the test set which is the point before the current date in the dataset which split from train set.

2. Only future data test

(a) Test by date

for each day from the point after the last date in the train dataset, we evaluate the model performance on it. the accuracy is compute on the number of correct predict fund across all funds.

(b) Test by fund

for each fund, we evaluate the model on the dataset after the last date in the train dataset to see the performance of model working on the particular fund. the accuracy is compute on the number of correct predict date from all future date in the dataset

5.3.2 Forecasting NAV of the next three months using Box-Jenkins method

Pre-processing data

In this approach, we use only the data in the last three years period because the older the data the less chance that it will correlate to the future data which we are going to forecast.

The data is then separated into two sets, training and testing set. The sets are created by sampling without replacement on the original data. The size of the testing set is 25% and the remaining 75% is the training set.

The training set is used to train, which allows the model to learn the parameters. The testing set will be used to measure the performance of the model and gather the testing result to evaluate whether the model is adequate enough to use for forecasting the NAV return in the next three months.

Model identification

Mostly the dataset is non-stationary time-series based on the Augmented Dickey Fuller test (ADF). The ADF score is too high, it is higher than all of the critical values and the p-value is also too high to reject the null hypothesis. A Different method is applied to make the dataset stationary. In most of the dataset we only differentiate one or two times to be stationary by doing the ADF test to visualise the result again. After getting the stationary dataset, we then plot the ACF and PACF to get potential lags to be the order p,q of the arima model.

Parameter Estimation

Then we apply the built-in function called Auto-Arima which is provided by Pmdarima to find the best coefficient order p,d,q of the model by setting the limit based on ACF and PACF lags that we plotted earlier. The function will calculate all of the possible order within the range of ACF/PACF lags and then calculate the maximum likelihood estimation which is AIC score (Akaike Information Criterion score) to select the order p,d,q that is the most efficient one for the ARIMA model

Diagnostic Checking

After getting the best coefficient order, we train the ARIMA model of the specific order in the training set then use that model to predict the NAV return in the same period as the testing set then compare with the testing set to calculate the model accuracy in the form of RMSE score. We also take residual from the prediction result to try to identify the signal whether it is white noise which means that all of the signal information in the time series has been harnessed by the model in order to make predictions. All that is left is the random fluctuations that cannot be modeled.

5.3.3 Forecasting NAV of the next three month using Long Short-Term Memory Neural Network

Preprocessing data

In this approach the data are the same as the first approach which is the NAV record of each mutual fund in the last three years and split the dataset into 70% of train set and 30% of test set. Due to the fact that this experiment is a time series problem. The dataset must be transformed into the Supervised learning first by using the observation in the last time lag as the input and the observation at the current time step as the output.

The modification is done by shifting all the values in the time series dataset down by a specified number places which in this case only one shift is enough. The pushed-down series will have a new position at the top with no value which the empty value slot will act as the beginning of the data then the two series are concatenated together ready for the Supervised learning.

From the first experiment, if the time-series is stationary it is easier for the model to forecast more accurate output. The trend can be removed with the same method which is differencing the time series and then added it back later in the prediction process to return the value into the original scale and calculate a comparable error score.

LSTM are also one type of the neural network that expects data to be within the scale of the activation function used by the network. In this case the default activation function for LSTM is the hyperbolic tangent function (tanh), which outputs values between -1 and 1 to be the preferred range of time series data. After finishing rescaling all the data, it has to invert back into the original scale in the prediction process so the result can be interpreted and a comparable score can be calculated.

LSTM Model Development

The Long Short-Term Memory network (LSTM) is a type of Recurrent Neural Network (RNN) that can learn and remember over the long sequences and not relying on a pre-specified window lagged observation as input.

By default, an LSTM layer in Keras maintains state between data within one batch. A batch of data is a fixed-sized number of rows from the training dataset that defines how many patterns to process before updating the weights of the network. State in the LSTM layer between batches is cleared by default, The LSTM layer expects input to be in a form of a matrix with three dimension: [samples, time steps, features] each of them are:

- Sample: The independent observations from the domain, normally rows of data.
- Time steps: The separate time steps of a given variable for a given observation.
- Feature: The separate measure observed at the time of observation.

In this experiment the NAV dataset is set to be simple to frame for the network as each time step remains still with the original sequence which is one separate sample, time step and feature. LSTM layers also require to set the batch size which is usually set in a smaller size of the total number of samples. Along with the number of epochs which defines how quickly the network learns the data and how often the weight of each neural network node are updated. Lasty, LSTM is required to declare the number of neurons or in the other word the number of memory units. In this experiment the number of neurons between 1 to 5 is sufficient to create the NAV forecasting model.

Once the network is specified, it must be compiled into an efficient symbolic representation and in the compiling process the loss function and optimization algorithm have to be define which in this experiment we using Mean Square Error function as a loss function because it closely matches with Root Mean Square Error(RMSE) that will be the main criteria to evaluate the accuracy of the model along with the efficient ADAM optimization algorithm. After the compiling process is done the model can be fitted to the training data.

LSTM Forecast

Once the LSTM is fit to the training data, it can be used to make forecasts. First we fit the model once on all of the training data, then predict each new time step one at a time from the test data. This approach is called a fixed approach which is not at the rate of satisfying. There is also another approach which tries to re-fit the model or update the model each time step of the test data as new observations from the test data are made available (dynamic approach).

5.3.4 LSTM result as the feature of classification network

We construct the experimental of the hybrid model in the same way as the classification network. But we add the result of the LSTM as the feature to feed in the classification network. the result of LSTM model is the future NAV in specific range of each fund. In this experimental, we use LSTM to predict next 90 from the current date. Then we apply geometric mean returns to those prediction result.So we get new feature call pred+3. Due to LSTM model is time consuming. so select only 3 date to demonstrate the performance of the model. so we add pred+3 feature to 2020-01-08, 2020-04-08 and 2020-07-08.

	updated	value
0	2020-01-08	11.901862
1	2020-01-09	11.904266
2	2020-01-10	11.905162
3	2020-01-13	11.906106
4	2020-01-14	11.906887

'1AM-TG'

	updated	value
0	2020-01-08	5.542261
1	2020-01-09	5.534031
2	2020-01-10	5.498820
3	2020-01-13	5.496976
4	2020-01-14	5.492177

'ABTED'

	updated	value
0	2020-01-08	10.008765
1	2020-01-09	10.008216
2	2020-01-10	10.008198
3	2020-01-13	10.010152
4	2020-01-14	10.011432

'ASP-DGOV-R'

	date	fund_code	nav	amount	set_index	return	return-3	return-6	beta-3	pred+3
0	2020-01-08	1AM-TG	12.0376	238626509	1559.27	0.000017	0.003770	0.004531	-0.001154	0.006079
1	2020-01-08	ABTED	5.5627	147815411	1559.27	-0.009403	-0.038026	-0.091661	0.001495	-0.016681
2	2020-01-08	ASP-THEQ	10.6198	99640124	1559.27	-0.017177	-0.047303	-0.051553	0.001391	0.023709

Figure 5.4: left figure is the sample of LSTM results and right figure is the dataset after add pred+3 feature

Chapter 6
Experimental Result

6.1 Classification network result

the accuracy and standard derivation of 1 hidden layer MLP models is show in the figure below.

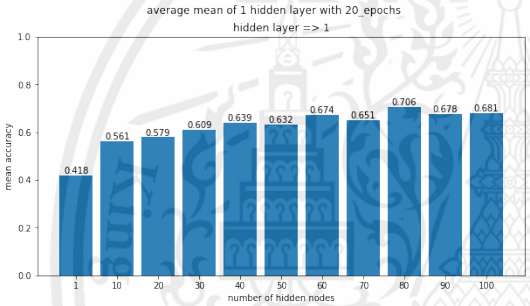

Figure 6.1: accuracy of 1 hidden layer model with 20 epochs

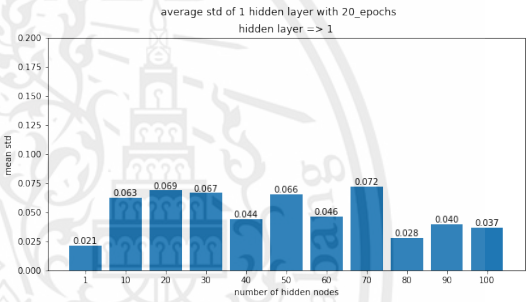

Figure 6.2: standard derivation of 1 hidden layer model with 20 epochs

from the figure it show the best number of hidden layer that give the maximum accuracy and minimize standard derivation is 80 nodes. with accuracy = 0.706 and = 0.028

the accuracy and standard derivation of 2 hidden layer MLP models is show in the figure below.

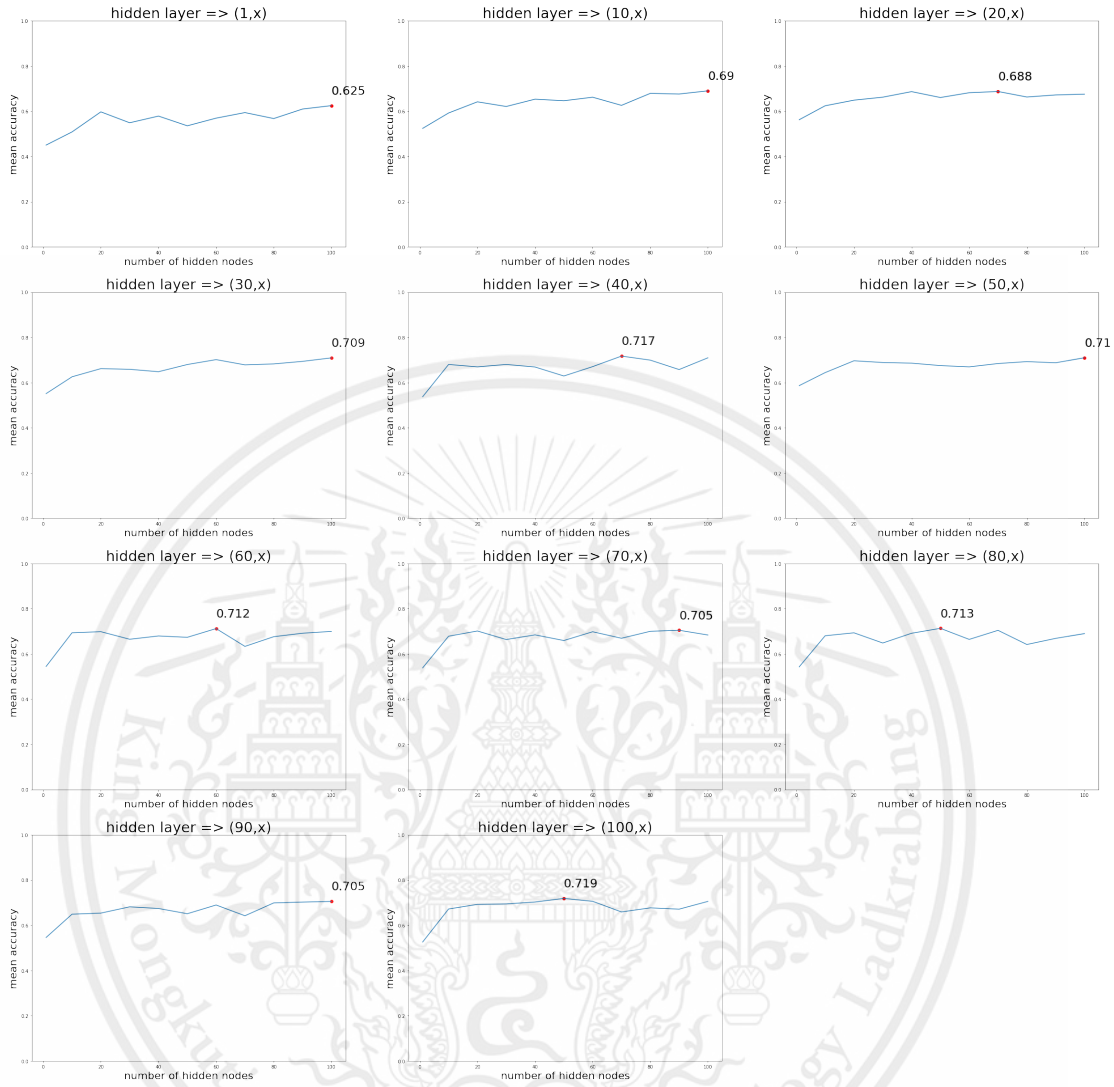

Figure 6.3: Accuracy of 2 hidden layer model with 20 epochs

each sub-figure is the accuracy of 2 hidden layer with different hidden nodes setting for example the upper left figure has 1 hidden nodes in first layer and others nodes is the value in hidden nodes test set. so the best accuracy over all 2 hidden-layer nodes is 0.718 with standard derivation equal to 0.13 From 2 hidden layer, It show that the increase of hidden layer is increasing accuracy but the model is more various. so we stop increasing the number of layer and focus on the number of epochs instead.

for the above figure, the model is train on 20 epochs. the result model with 50 epochs for 1 and 2 hidden layer is show below

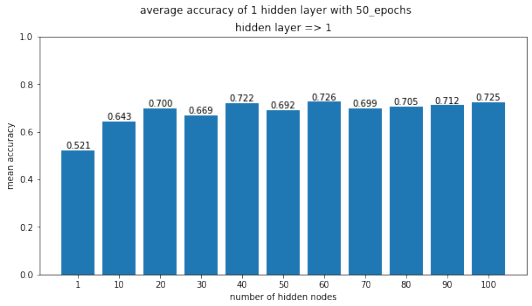

Figure 6.4: accuracy of 1 hidden layer model with 50 epochs

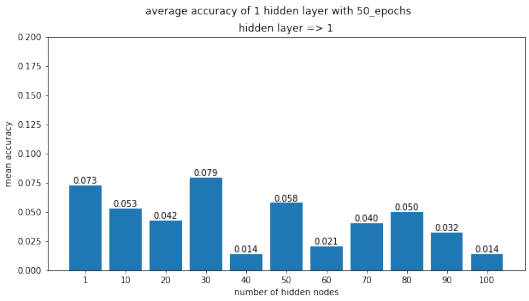

Figure 6.5: standard derivation of 1 hidden layer model with 50 epochs

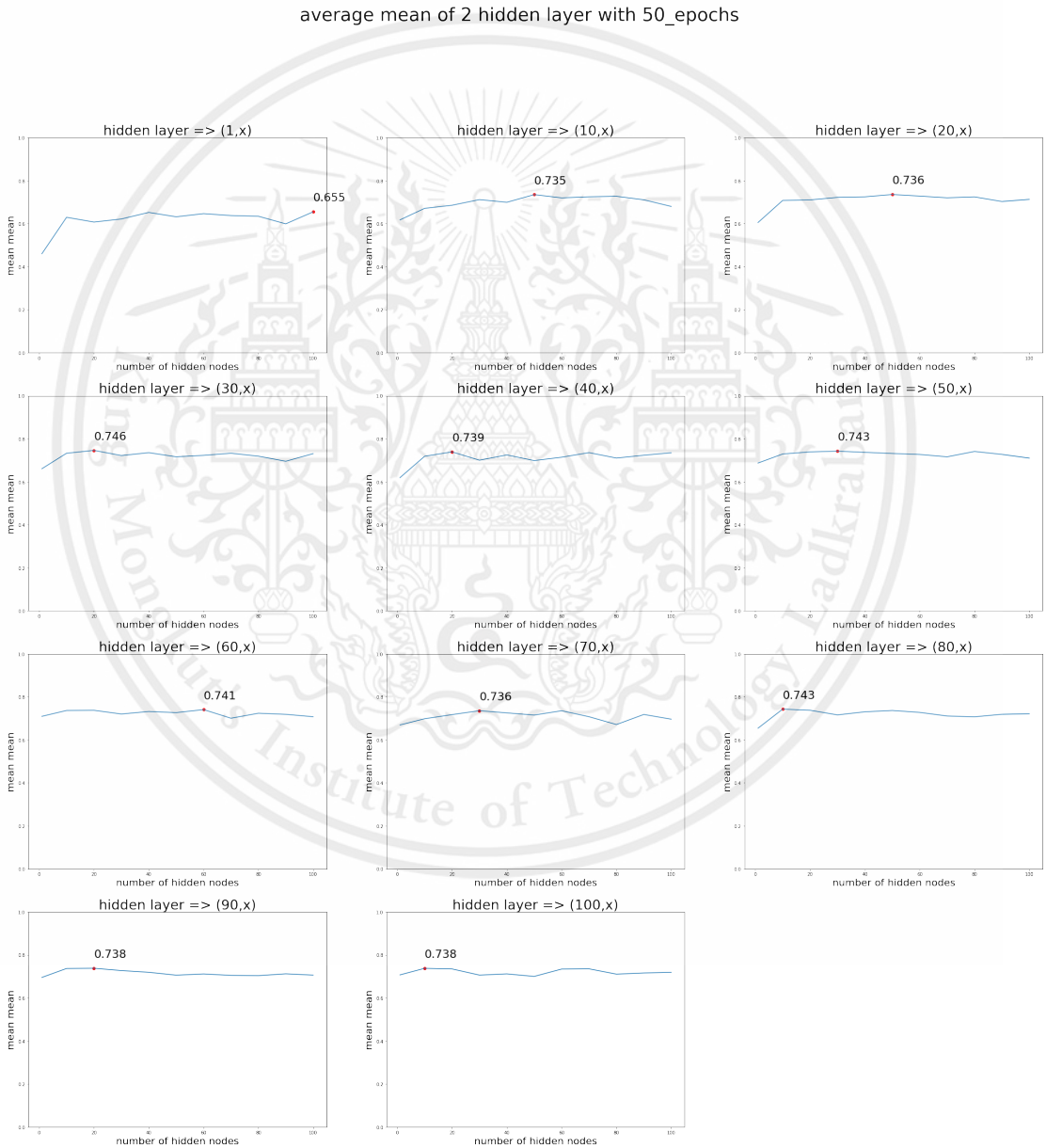

Figure 6.6: Accuracy of 2 hidden layer model with 50 epochs

With 50 epochs, the accuracy of the models are increase. the best configuration of 1 hidden layer is 60 nodes with accuracy equal 0.726 and std equal 0.079 and 2 hidden layers is 30 nodes in first layer and 20 in second layer with accuracy equal 0.746 and std equal 0.121. So the accuracy is not large increase so stop test on others number of epochs and select the best model so far to test on detail which is 2 hidden layers with (30,20) nodes setting.

After we got the base model. we looking forward to the validation test on that model and the result is show below:

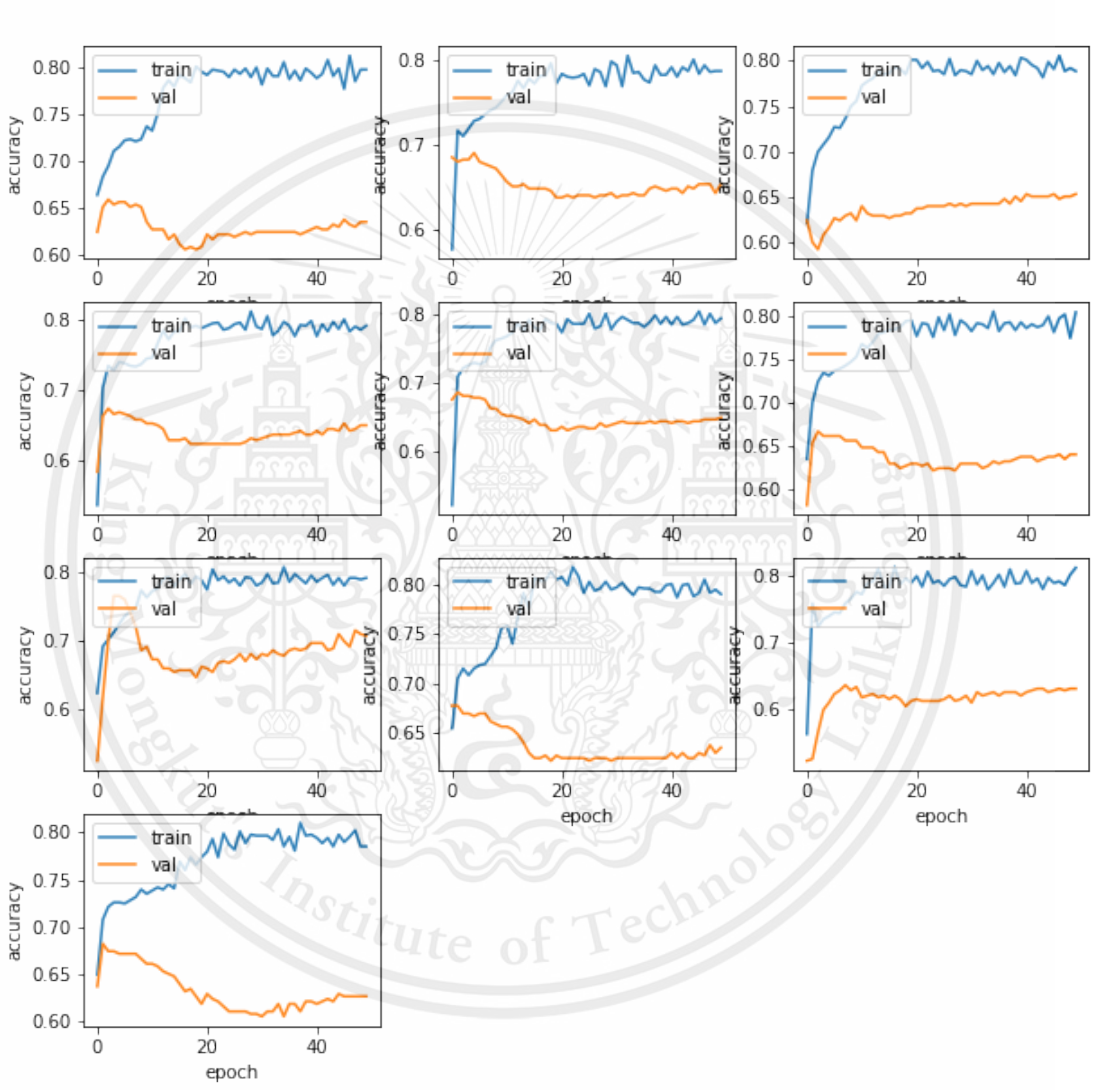

Figure 6.7: validate accuracy plot

from 10 models validate plot, we use 30 percent of train set as validate set. the gap between train and validate is around 20 percent which appear the overfit problem.So we apply different reducing overfit method to the model

Result of reducing overfit in model

Early stopping

In our problem, we set the model will stop training when the validate accuracy of the epochs is not decrease.

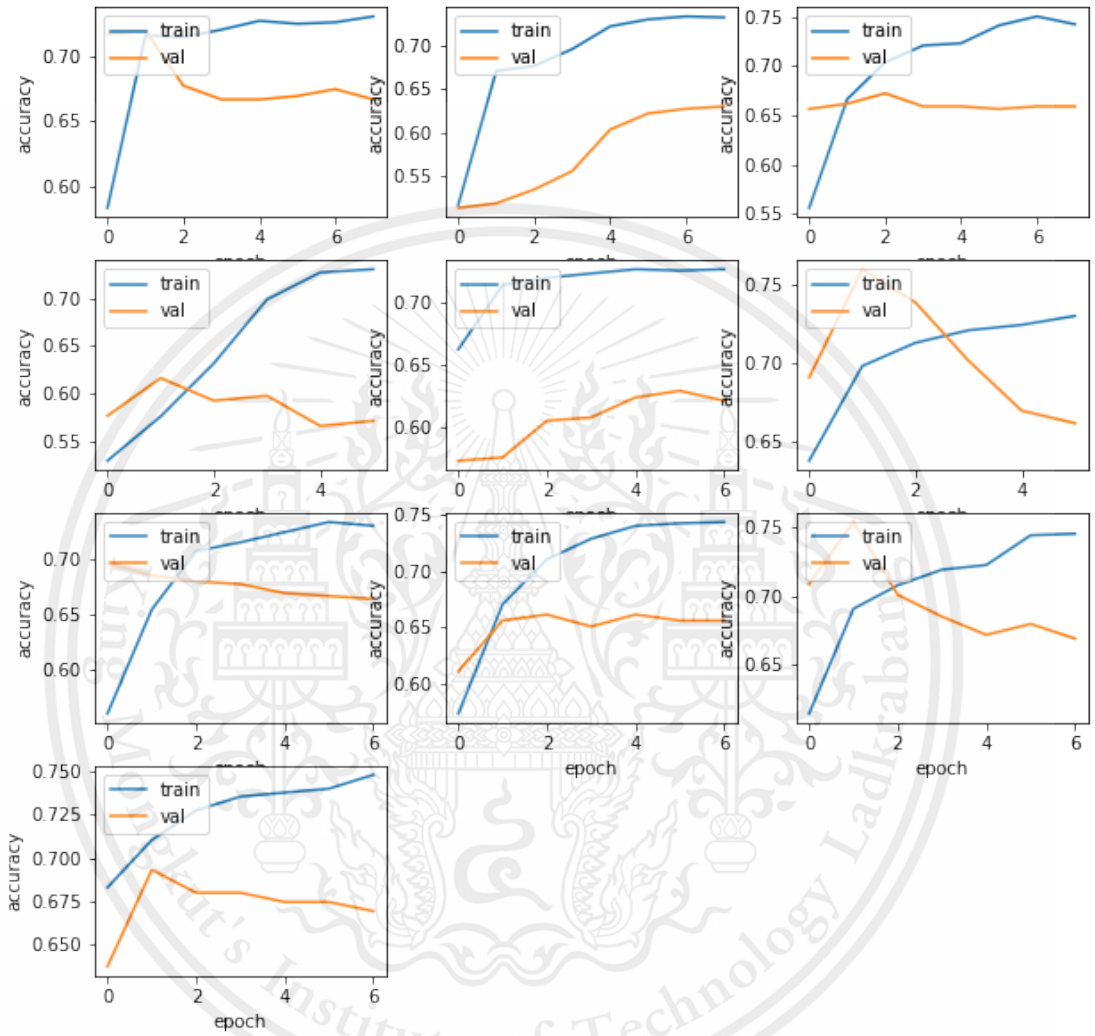

Figure 6.8: validate accuracy with early stopping

l1 and l2 regularization

We try to use both l1 and l2 at the same time in the model like the Elastic Net which is modern and effective linear regression methods. we select l1 weight to be 0.5 and l2 weight is 0.9

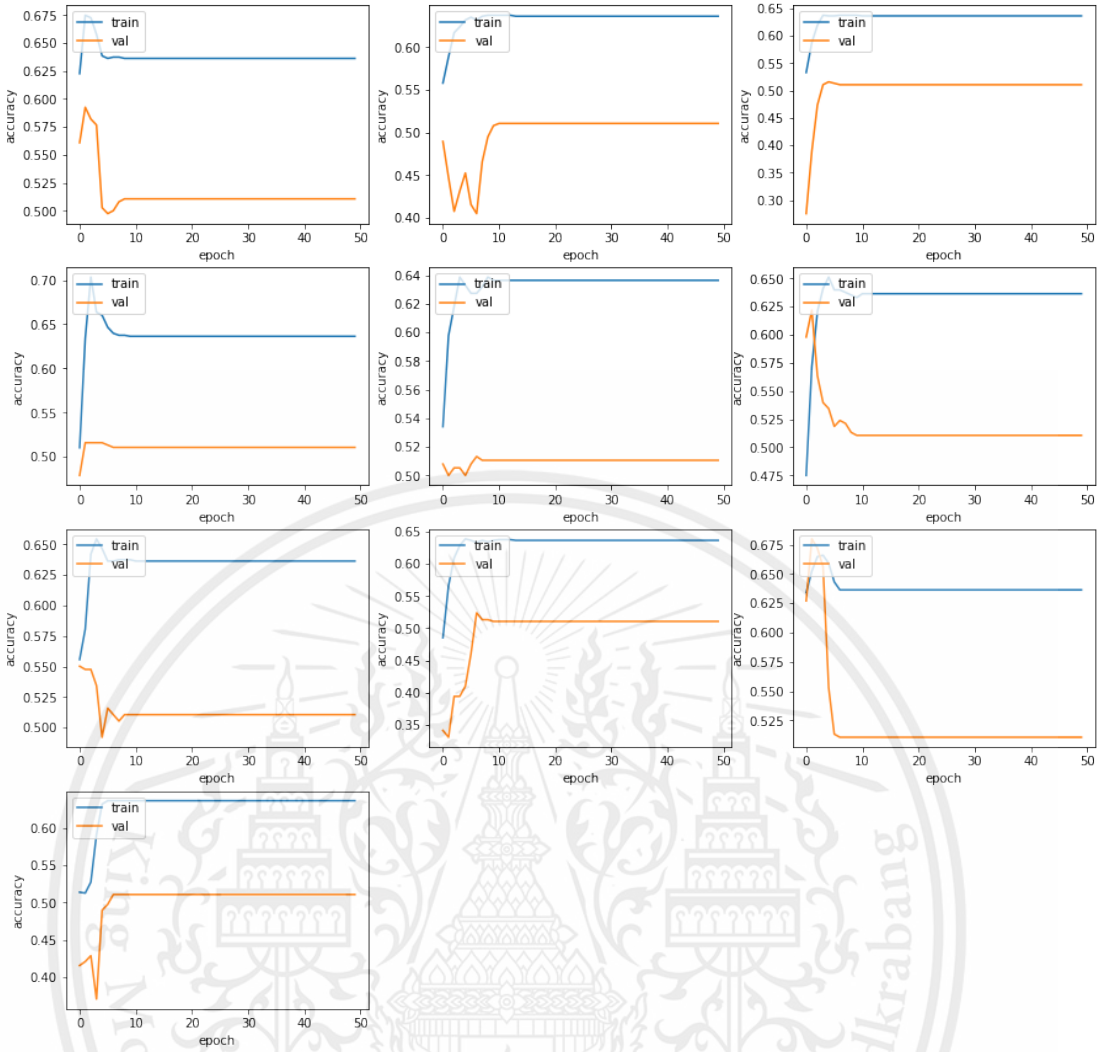

Figure 6.9: validate accuracy with l1 and l2 legalization

this configuration make weight initialize not affect to the result all models has accuracy equal to 0.59

drop out

From the result of GridSearchCV, the best dropout_rate is 0.4.So we add dropout layer to model.

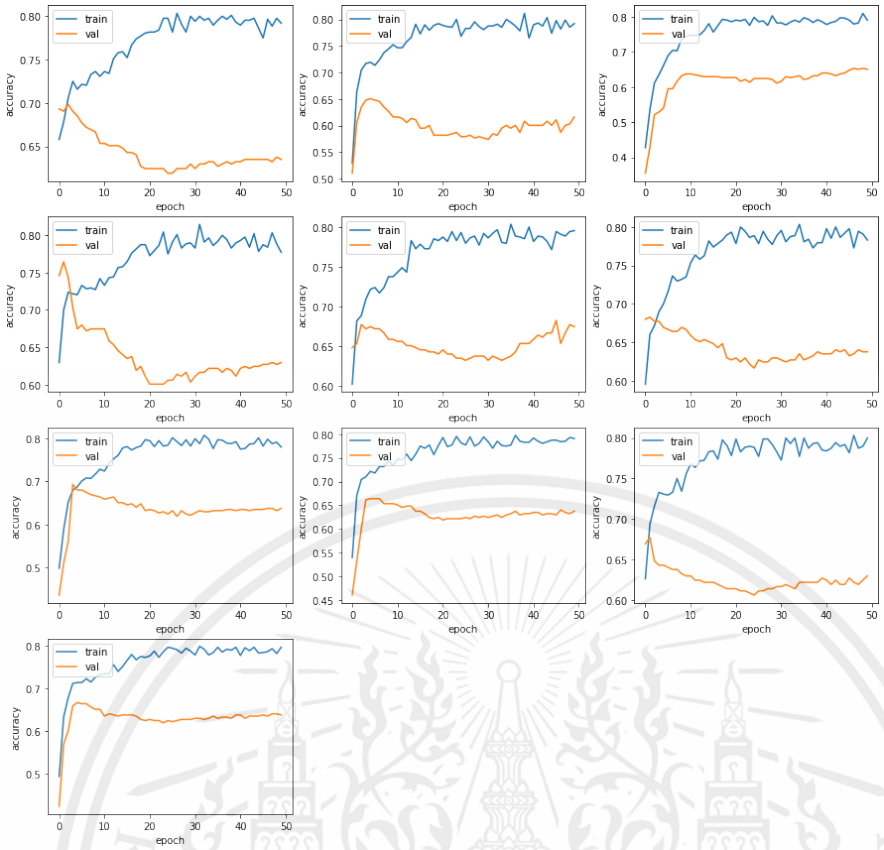

Figure 6.10: validate accuracy with drop out

the accuracy is increase to 0.753 and is reduce to 0.007

From all reducing overfit method show, the method that have smallest mean average gap between train and validate is l1 and l2 regularization. So we select the third model of this method for further test.

6.1.1 Testing result

Day test result

The testing on day test of the model show that model is suit for the normal situation of the market. it not flexible to the change of the market. Most of predictions are not correct at the same day show the change of the market effect to the model

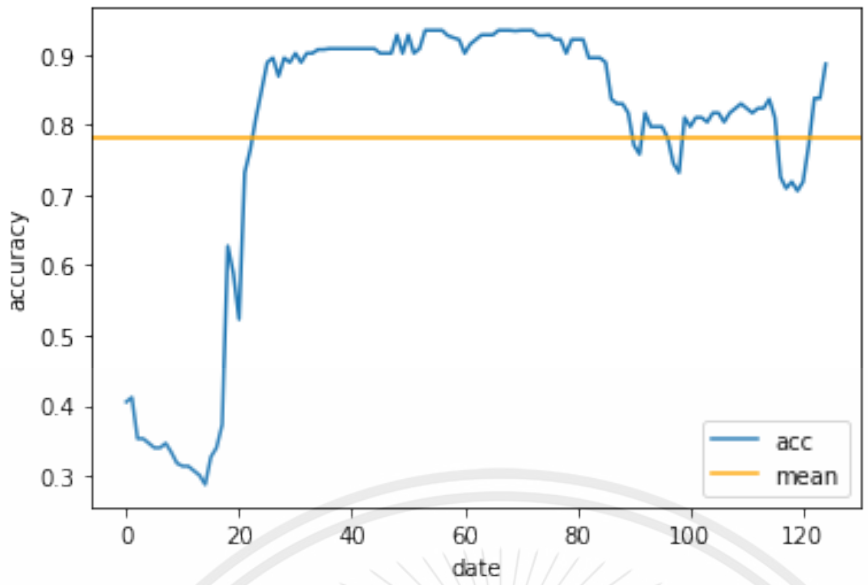

Figure 6.11: Day test result

Funds test result

From the funds test it show model is suit for specific funds even thought we already scope the condition of funds.

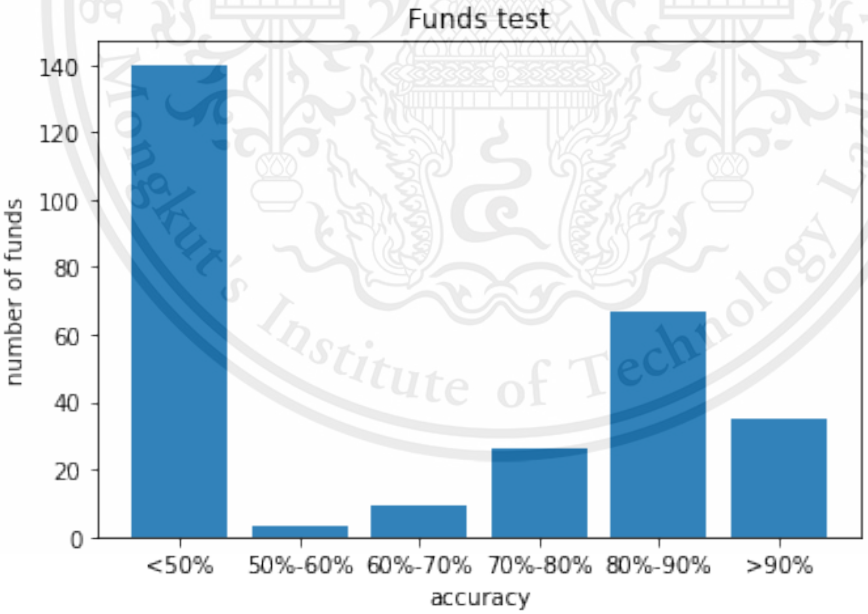

Figure 6.12: Funds test result

6.2 Result of Forecasting NAV of the next three months using Box-Jenkins method

This result is gathered by applying the experimental approach to one of the mutual fund dataset which is nav of PRINCIPAL GFIXED that contains 626 rows of data. The observed result is shown below.

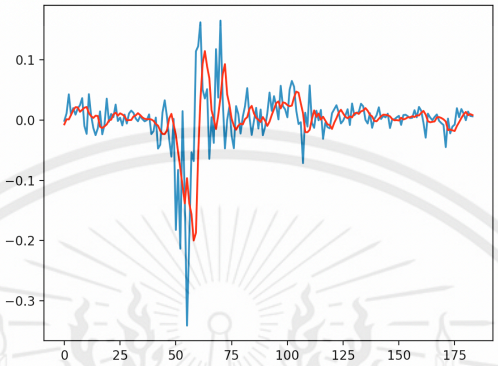

Figure 6.13: the prediction result. blue are the testing set data red are prediction data which the y-axis is NAV return

Table 6.1: Error score

Error criteria	Score
Mean of Residual	0.000151
MSE	0.027726
RMSE	0.002487

from the prediction result graph, we can see that the prediction result might not give the exact return of NAV but it follows the trend of the testing set graph.so we can conclude that the prediction is good enough to define a future trend whether is higher or lower than the present value. For the error score the RMSE and MSE score are very low and close to zero which can be conclude that the model is quite accurate with the very low value of mean of residual as a one criteria to be white noise which we will explain below.

From the figure 6.14 - 6.16, we can see the significance sign of residual to be the white noise which means that the model is quite accurate.In figure 6.13 we plotted the residual graph of the model we can visualise that the time-series is stationary with the mean is very closed to zero but the variance still be varied in some point and not equally to zero. In figure 6.13 it is the auto-correlation plot of the residual which most of the lag

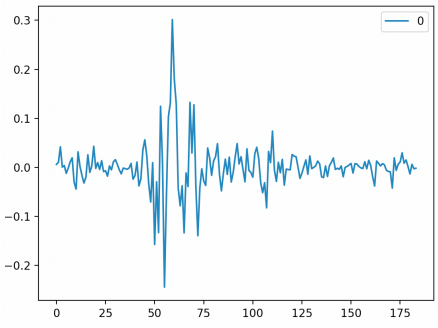

Figure 6.14: the residual plot from the prediction result

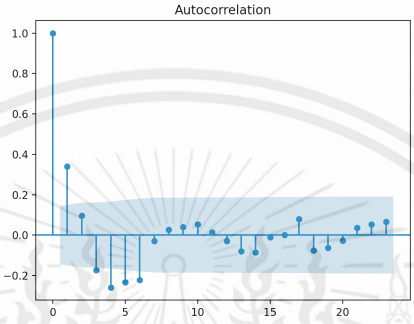

Figure 6.15: the auto-correlation function plot of residual

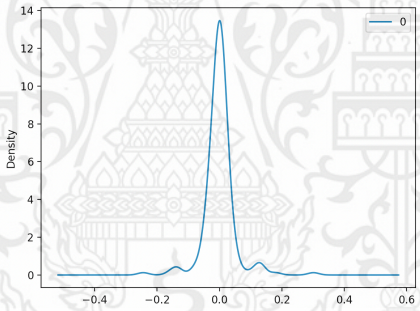

Figure 6.16: the density of residual appear to be in Gaussian form

is lesser than the boundary which means it's not correlated to each other. This satisfies the white noise criteria. The last figure 6.15 which is the density plot of residuals which we can visualize the curve that appear to be in the Gaussian equation form which also indicates as one of the white noise criteria. Most of the signal information in the time series has been harnessed by the model in order to make predictions. All that is left is the random fluctuations that cannot be modeled.

From all the experimental results that we mentioned, we can conclude that the model is accurate if we can get the best coefficients in order to be the ARIMA model so in the future the selection order criteria should be more precise in order to get that coefficient and try to apply some other model to compare the performance rate with this experiment.

6.3 Result of Forecasting NAV of the next three month using Long Short-Term Memory Network

After the training model process is done, Loss has been calculated by the loss function and then plotted to observe the training performance then comparing the result in each epoch along in the graph which in this scenario the epoch was set to be 500 and contain 4 hidden nodes.The result is shown below.

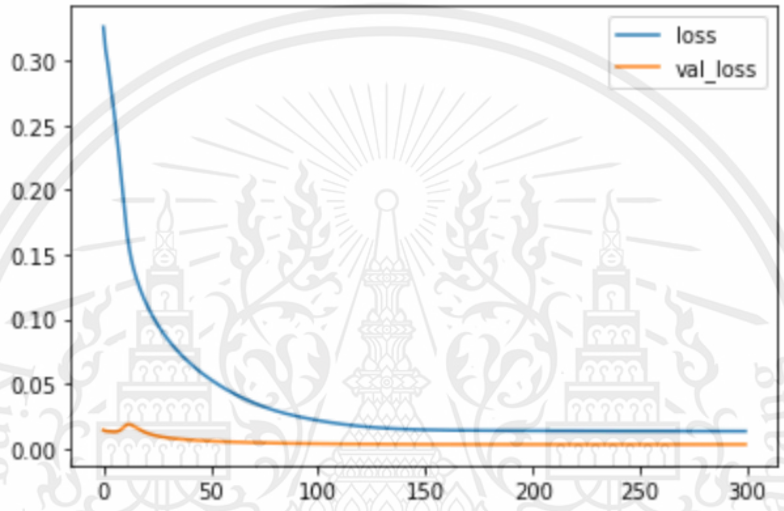

Figure 6.17: Plot on model Loss on training and validation datasets

From the plot of loss and validation loss, it can indicate that the model has comparable performance on both train and validation datasets and From the graph of the loss pattern it indicated that the model tends to be overfitting. Overfitting refers to a model that has learned the training dataset too well, including the statistical noise or random fluctuations in the training dataset. The problem with overfitting is the fact that the more specialized the model becomes to fit the data the less well of its capability to generalize to new data and resulting in increasing generalization error that means the model requires more tuning process or increases the number of training data that feed to the model.

The prediction result is gathered from conducting an experiment on one of the mutual fund NAV dataset which in this case are SI-TDEX (ThaiDEX SET50 ETF) records in the last three year. The dataset contains 670 rows of data which split into 469 rows of training set and 201 rows of testing set in the ratio of 70/30. The observed result is shown below

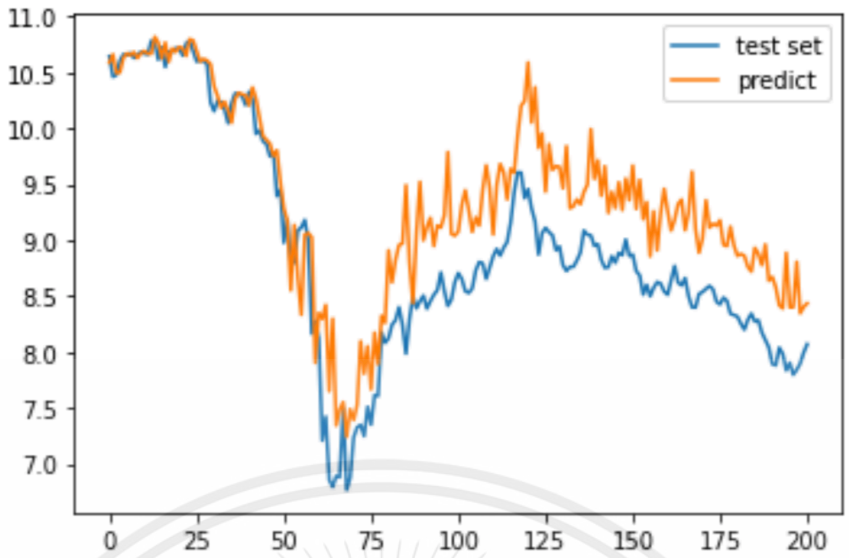

Figure 6.18: the prediction result compare to the testing set

From the plot of the prediction result, the model predict the test set with the Root Mean Square Error score(RMSE) of 0.484 and the prediction can capture mostly the trend of NAV in the testing set. The model got quite accurate in the beginning of the prediction process but when the model reached around 50% of the testing set, the prediction went higher than the expected value but also still remained capturing the trend of the testing set and ended up with low data loss calculated from Root Mean Square Error function. The prediction result is still too overfitted due the fact that the input that was feeding into network is not in the proper sequential form and number of some hyper-parameters like number of neuron and epoch is not suitable for the dataset and using the Multi-step forecast and LSTM model tuning.

6.3.1 Result of the combination of classification network and LSTM network

we only compare the base model with pred+3 and without pred+3 dataset. Due to the range of training data is too short because LSTM require lot of time to adjust and train. So the model that train with perd+3 get accuracy equal to 0.87 with equal to 0.004 and model with out pred+3 get accuracy equal to 0.88 with equal to 0.005. The result appear good but only for the train with out future data test. It show that in short term dataset the result of LSTM network not improve the performance of the classification network

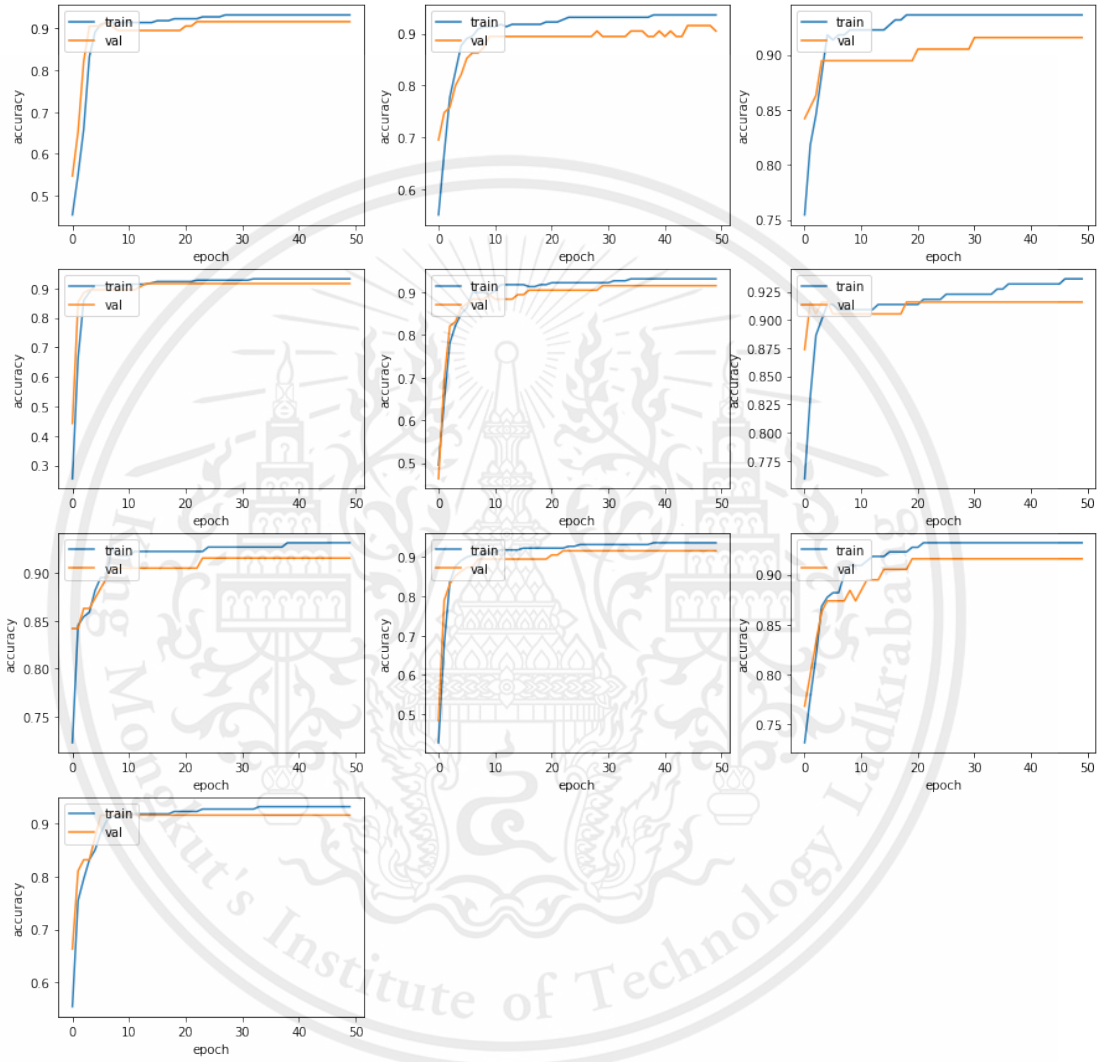

Figure 6.19: validate accuracy of dataset with out pred+3

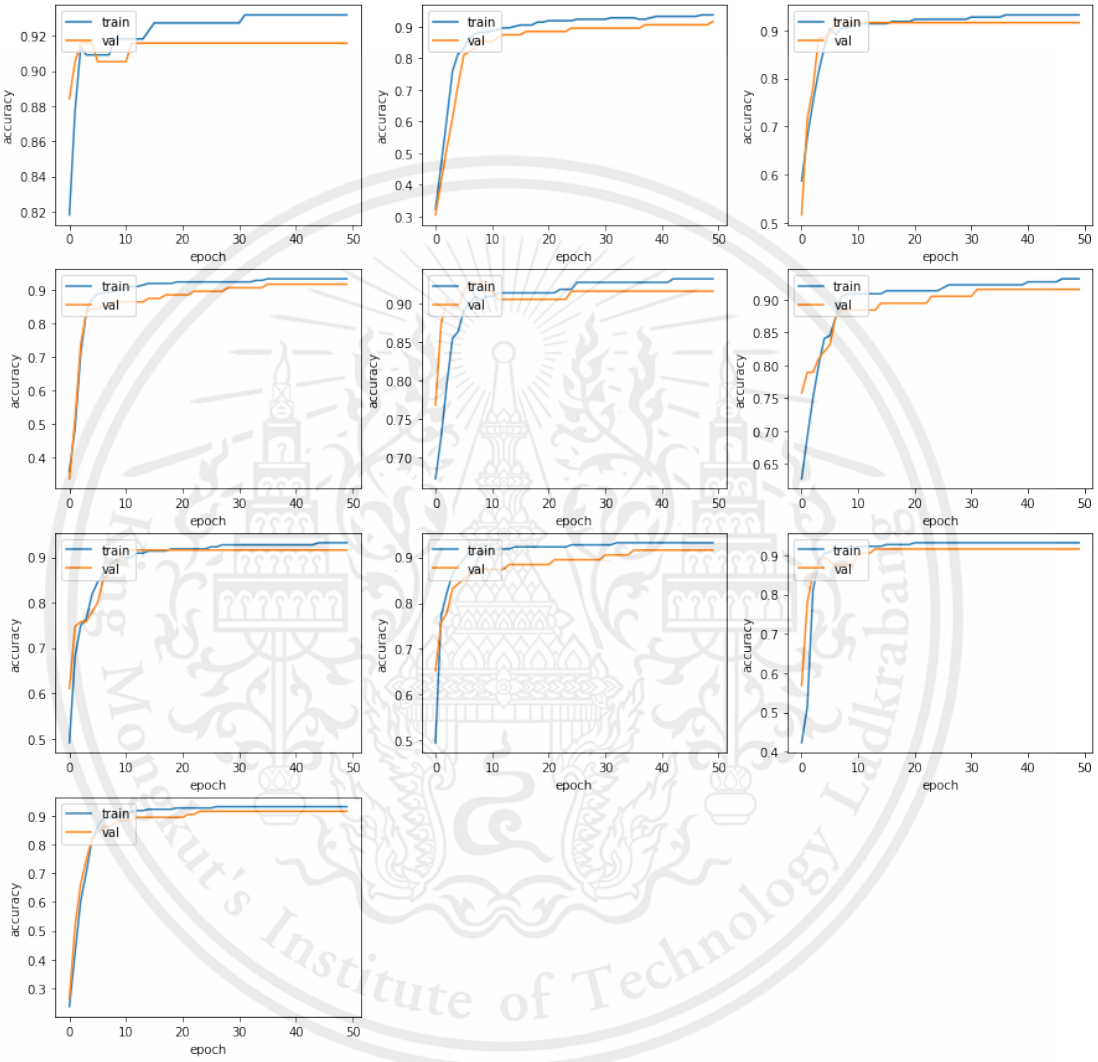

Figure 6.20: validate accuracy of dataset with pred+3

Chapter 7
Conclusion

The objective of this research is to apply the machine learning algorithm to predict the future performance of Thai mutual funds.And from several research, we found that the buy and hold strategy is suit for investing in mutual funds. So we decide to predict the future performance of Thai mutual funds in 3 months.

We design to achieve the objective in 3 approach following below:

1. Classification future performance using neural network In this approach, we turn the problem to be classify to buy or not buy and use traditional neural network to predict the performance.

2. Forecasting NAV using long short-term Memory Network This approach use benefit of time-series dataset. data in previous time-frame are effect to the result of forecasting.

3. Combination of first and second approach Using the result of LSTM model as the feature of classification network to predict future performance

If we focus only the NAV time-series, LSTM is work well with RMSE score equal to 0.484 . The model is quite accurate and can be improve tuning parameter and increasing training set. But In the real world, it have others factor that effect to the performance of funds.

From the best classification model (less sensitive to out-sample data) which use the financial ratio get accuracy from test around 59 percent to correct predict. and when we test the model in out of sample data, it show that model can not react to the change of the market well and in each funds test it show that around 140 funds is not suit for the model, They have accuracy under 50 percent even through we scope the funds before.

For the combination of LSTM and classification network, LSTM require lot of time to train and adjust. and from the experimental, apply LSTM in the short term not improve the performance of the model.

For conclusion, This research show that it potential to apply machine learning algorithm to predict the future performance of mutual funds but it require more data to

train the model to tolerant to the variance of the market and also have to keep update the technique to increase accuracy and reducing the gap between train-test and validate-test

Bibliography

[1] A. ORRE, and A. PERS, Classifying Mutual Funds Based on Relative Performance, Kth Royal Institute of Technology, Sept. 2017, kth.diva-portal.org/smash/get/diva2:1196404/FULLTEXT01.pdf.

[2] E. Priyadarshini, A. Govindarajan , E. Sharon Preethi, A Hybrid Approach for Forecasting the Net Asset Values, Inter National Journal of Pure and Applied Mathematics, 2017, acadpubl.eu/jsi/2017-113-pp/articles/12/24.pdf.

[3] A. Adebiyi,A. Adewumi, Stock Price Prediction Using the ARIMA Model . University of KwaZulu-Natal Durban, 2017, doi:10.1109/UKSim.2014.67

[4] A. Wadi,M. Almasarweh, A. Alsaraireh, Predicting Closed Price Time Series Data Using ARIMA Model . The University of Jordan, 2018, doi:10.5539/mas.v12n11p181

[5] S. Mcnally, J. Roche, and S. Caton, "Predicting the price of Bitcoin using Machine Learning", In parallel, Distributed and Network-based Processing(2018), 2018 26th Euromicro International Conference, Cambridge, UK, March 2018, pp.339-343

[6] A. Hayes. "Mutual Fund Definition." Investopedia, Investopedia, 14 Dec. 2020, www.investopedia.com/terms/m/mutualfund.asp.

[7] R.J. Hyndman, G. Athanasopoulos, Forecasting: principles and practice, 2nd edition,2018 , OTexts: Melbourne, Australia. OTexts.com/fpp2

[8] J. Brownlee, What is time series forecasting?, Machinelearningmastery , 2 Dec 2016, https://machinelearningmastery.com/time-series-forecasting

[9] Box-Jenkins Models , Engineering Statistic Handbook, NIST SEMATECH, https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc445.htm

[10] "Machine Learning." Wikipedia, Wikimedia Foundation, 14 Dec. 2020, en.wikipedia.org/wiki/Machine_learning.

[11] “Eigenvalues and Eigenvectors.” Wikipedia, Wikimedia Foundation, 10 Dec. 2020, en.wikipedia.org/wiki/Eigenvalues_and _eigenvectors.

[12] S. Hochreiter, and J. Schmidhuber, ”Long short term memory,” Neural computation, vol. 9, no. 8, pp. 1735-1780, Nov. 1997.

[13] C. Olah, ”Understanding LSTM Networks”, 27 Aug. 2015, https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[14] ”Model Fit: Underfitting vs. Overfitting”, 19 Dec. 2015, https://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html