



School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - GROUP ASSESSMENT

Unit of Study: COMP5318 Machine Learning and Data Mining

Assignment name: COMP5318 Assignment 2

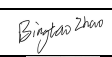
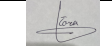

Tutorial time: 5pm-6pm 8pm-9pm Tutor name: kunzeWang Thomas Selvaraj Henry Weld

DECLARATION

We the undersigned declare that we have read and understood the *University of Sydney Student Plagiarism: Coursework Policy and Procedure*, and except where specifically acknowledged, the work contained in this assignment/project is our own work, and has not been copied from other sources or been previously submitted for award or assessment.

We understand that failure to comply with the *Student Plagiarism: Coursework Policy and Procedure* can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

We realise that we may be asked to identify those portions of the work contributed by each of us and required to demonstrate our individual knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Project team members				
Student name	Student ID	Participated	Agree to share	Signature
1. Bingtao Zhao	490041116	Yes / No <input checked="" type="checkbox"/> / <input type="checkbox"/>	Yes / No <input checked="" type="checkbox"/> / <input type="checkbox"/>	
2. Tara Ghazanfari	470450169	Yes / No <input checked="" type="checkbox"/> / <input type="checkbox"/>	Yes / No <input checked="" type="checkbox"/> / <input type="checkbox"/>	
3. Fengrui Yang	500692202	Yes / No <input checked="" type="checkbox"/> / <input type="checkbox"/>	Yes / No <input checked="" type="checkbox"/> / <input type="checkbox"/>	
4.		Yes / No <input type="checkbox"/> / <input type="checkbox"/>	Yes / No <input type="checkbox"/> / <input type="checkbox"/>	
5.		Yes / No <input type="checkbox"/> / <input type="checkbox"/>	Yes / No <input type="checkbox"/> / <input type="checkbox"/>	
6.		Yes / No <input type="checkbox"/> / <input type="checkbox"/>	Yes / No <input type="checkbox"/> / <input type="checkbox"/>	
7.		Yes / No <input type="checkbox"/> / <input type="checkbox"/>	Yes / No <input type="checkbox"/> / <input type="checkbox"/>	
8.		Yes / No <input type="checkbox"/> / <input type="checkbox"/>	Yes / No <input type="checkbox"/> / <input type="checkbox"/>	
9.		Yes / No <input type="checkbox"/> / <input type="checkbox"/>	Yes / No <input type="checkbox"/> / <input type="checkbox"/>	
10.		Yes / No <input type="checkbox"/> / <input type="checkbox"/>	Yes / No <input type="checkbox"/> / <input type="checkbox"/>	

COMP5318 Assignment 2 Report

Group 23

November 17, 2020

Abstract

Stock market is a fundamental part of every economy. Businesses and industries can grow with the help of shares that stock market issues. The Stock price prediction has been one of the most attractive problems among many investor communities and certainly is one of the most important facets in modern financial studies. In this paper three different methods are used to predict the close price of New York Stock Exchange. We perform experiments by applying ARIMA, Lasso Regression and LSTM machine learning models on this data. Then compare *rmse* of these methods to obtain the best performing algorithm. Our experiments indicate that all the methods are used perform really good, however, lasso model performs the best in terms of accuracy of predicting short term price.

1 Introduction

Stocks may be one of the most important investment assets in the world, seeking rules of stock markets is a major research topic in investors and institutional researchers, especially for technical analyst. Technical analysis refers to the methods of making stocks and all financial derivatives trading decisions based on market behavior as the research object, judging the market trend and following the cyclical changes of the trend. Technical analysis believes that market behavior tolerates and digests all information, prices fluctuate in a trend, and history will repeat itself. Although technical analysis is considered by academia to be inaccurate and without sufficient theoretical basis, on the other hand, it can help to prove the efficient market hypothesis and random walk theory which think that the market cannot be predicted by historical data.

Thus, our report studies the feasibility of predicting future stock price from historical data by using the machine learning method which may reveal the connection that the image analysis and trend analysis used in traditional technical analysis failed to find. The dataset we used is New York Stock Exchange (<http://www.kaggle.com/dgawlik/nyse>), which is the world's largest stock exchange by market capitalization and is considered to be the best example of Semi-Strong Form Efficiency (Investors cannot use intelligence analysis results to make stock price predictions). We would apply three machine learning and data mining methods to fit the original data and try to predict stock price, the results will reflect the market efficiency and give experimental support to one of Technical analysis or Efficient market hypothesis. To simplify data and compare models, we choose data of Microsoft (MSFT) as our original data set. Here is the stock price data of MSFT.



Figure 1: MSFT Close Price History

2 Literature review

2.1 ARIMA

Time series models are already variously used in financial studies, as early as 1982, Engle suggested the ARCH (autoregressive conditional heteroscedasticity) model that is famous in financial analysts[1], then the general form of ARCH, GARCH (General ARCH) model is proposed by Bollerslev[2], EWMA (Exponentially Weighted Moving Average) which is basically a non-stationary GARCH(1, 1) model was used by RiskMetrics from JP Morgan in 1996[3]. In recent years, ARIMA model is consider to be robust and efficient in time series forecasting (especially short-term prediction) [4], which has been widely used in the economic and financial industry. This report introduced the process of building an ARIMA model and tuning hyperparameter for short-term stock price forecasting and long-term prediction, and exploring the validation of financial theory from results instead of introducing a new model.

2.2 Lasso Regression

Regression is one of the most robust statistical model that is deployed in business and marketing researches. In the research was done by Aseervatham et al.[5] it concluded that ridge logistic regression and Support Vector Machine are performing the same. Setiawan et al.[6] use lasso techniques to predict Indonesian stock price. They compared the result of lasso model with linear regression and showed that lasso producing more accurate results. In our work we train lasso model on NEW York Stock data and compare the accuracy of that with SLTM and ARIMA models.

Other different methods also have been used to forecast stock price of other countries and region. For example, Yoo et al.[7] considered different events in their work and focus on predicting stocks market. They discovered that combining that event information with forecasting models can help significantly with more accurate prediction. A unique method is used by Ticknor [8] to forecast the financial market behavior. A Bayesian regularized artificial neural network is proposed in their work. Flexible neural tree approach is presented by Yuehui Chen et al.[9] to train on organized representation of stock data. They optimized their method by deploying genetic programming.

2.3 LSTM

Long and short-term memory (LSTM) is a recurrent neural network (RNN) architecture commonly used in the field of deep learning. It solves many tasks that previously cannot be solved by recurrent neural networks[10]. Unlike standard feedforward neural networks, LSTM has feedback connections. It can process not only a single data point (such as an image), but also an entire data sequence (such as voice or video input).

LSTM is mainly to solve the problem of gradient disappearance and gradient explosion during long sequence training. Simply put, compared to ordinary RNNs, LSTM can perform better in longer sequences. In 2009, the artificial neural network model built with LSTM won the ICDAR handwriting recognition competition. LSTM is also commonly used for autonomous speech recognition. In 2013, the TIMIT natural speech database was used to achieve a record of 17.7% error rate. As a nonlinear model, LSTM can be used as a complex nonlinear unit to construct larger deep neural networks[11].

There are many versions of LSTM, one of the important version is GRU (Gated Recurrent Unit). According to Google's test, the most important one in LSTM is Forget gate, followed by Input gate, and last is Output gate[12].

3 Methods

3.1 ARIMA

ARIMA (Auto-Regressive Integrated Moving Averages) is a very popular statistical method for time series forecasting, which has shown efficient capability to generate short-term forecasts with assumptions as follows:

1. The data series is stationary, which means that the mean and variance should not change over time. The logarithmic transformation or difference can make the sequence stationary.
2. The input data must be a univariate series, because ARIMA uses past values to predict future values.

ARIMA has three components which would be fine-tuned in experiment: AR (autoregressive term), I (differential term) and MA (moving average term), expressed as follows:

$$X_t = \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \dots + \alpha_p X_{t-p} + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \dots + \beta_q \varepsilon_{t-q} \quad (1)$$

Where X is the original data to fit and consists of past values (X_{t-p}) that predict the next value (X_t), ε_t is the random error at time t , α is the coefficient of AR with length of p , β is the coefficient of MA (the number of past forecast errors when predicting future values) with length of q , I is not shown in equation but is denoted as d , expressed as differential term (the number of differencing operations performed on the sequence). ARIMA is suitable for our dataset which contains transaction data every business day met the requirement of time series since only endogenous variables are needed without no other exogenous variables, we only need to show the stationary of time series data after differencing, which means the data is not trending or periodic, we will discuss this later. If we assume the data are self-correlated (momentum exists in stock market), ARIMA model would fit the data well and make good predictions.

In this case, we are going to use the close prices of past p days to predict the price of the next day, which meets the requirement of AR term then we use pre-processing techniques for treatment.

Firstly, check and drop missing data and then use Savitzky-Golay filter for data smoothing which is the need of stationary assumption.

Savitzky-Golay filter is widely used in data stream smoothing and denoising. It is a filtering method based on local polynomial least squares fitting in the time domain. The biggest feature of this kind of filter is that it can keep the shape and width of the signal unchanged while filtering noise. Its operation on the signal is to perform polynomial fitting on the data in *window length* in the time domain. From the frequency domain point of view, this kind of fitting actually passes the low-frequency data while filtering out the high-frequency data. For simplify, we only use Microsoft ('MSFT') stock to fit our model, and use first 1500 trading days for fitting and the rest as test data.

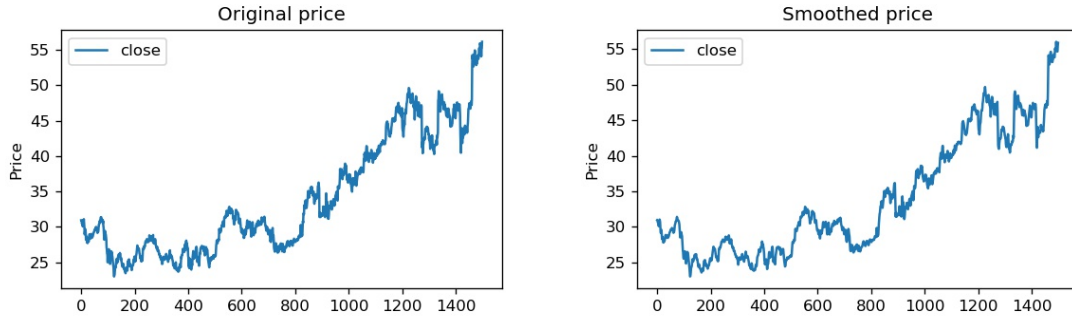


Figure 2: Original price vs Smoothed price of 'MSFT'

Figure 2 shows the effect of filter, the mid-to-long term trends of data does not change but filter out high frequency data which may result in the miss-fitting of model. Thus, we improve the accuracy of model in predict trends with the cost of accuracy on exact value.

3.2 Lasso regression

The LASSO (least absolute shrinkage and selection operator) is one of the regression model that can carry out feature selection as well as regularization in order to improve the performance of statistical models in machine learning. This method is a modification of the least square method of linear regression. In ordinary linear regression model the coefficients w_i are used to minimize the residual sum of squares between the actual predictors and the predictors that are produced by linear model [13]. The final form of objective function of linear regression can be defined as follow:

$$\frac{1}{N} \|Xw - Y\|_2^2 \quad (2)$$

Where X is the predictor variable, Y is the response variable, N is the numbers of instances and w is coefficient vector. In lasso regression sparse coefficients are estimated. It cause less predictors and it is specially important in obtaining the appropriate number of non zero variables under of certain constraints. These coefficients

can be obtained by the Lagrange equation that minimize the sum of the squares residual, with certain circumstances. $\|w\|_1 \leq t$, here t is the pre-specified quantity that handle the amount of shrinkage in estimating of coefficients[14]. When w_i is the least square estimator and $t_0 = \|w\|_1^1$, Then $t < t_0$ and least square method solution will reduce to zero, and causes some coefficients to reduce to an exact zero. The objective function for lasso can be written as follow:

$$\frac{1}{2N} \|Xw - Y\|_2^2 + \alpha \|w\|_1, \alpha \geq 0 \quad (3)$$

Where α is lasso parameter. $\alpha \|w\|_1^1$ explains the minimization of the least square penalty .

We expect lasso performs well for New York Stock Exchange data. Because lasso performs very well when there are small numbers of predictor variable compering to the size of sample. For lasso regression we use the same stock(MSFT) as ARIMA method. First 1500 trading days are deployed to fit lasso model. Each of MSFT samples are made of information including low price, high price, opening price, close price, and trading volume. We are predicting close price. So we chose close as the response variable and all the other features as the predictor variables. The training data set that we select are the first 1500 of the samples, while the test data set consisted the remaining of the samples. Train set then split to make a validation set. First, 1200 sample are used to train our lasso model and next 300 sample are utilized for validation purpose. The data was originally clean with no missing value. No preprocessing has done on the data. To fit lasso model we used

3.3 LSTM

LSTM is a logic loop. Compared with the Recurrent Neural Network(RNN) that stores two states, LSTM can store four states, the current and previous values of the output value, and the current and previous values of the memory neuron state. They all have three gates: input gate, output gate and forget gate. At the same time, they also have regular input. The function of these three gates is to protect information by prohibiting or allowing the flow of information. The difference of construction between RNN and LSTM have show in below figure:

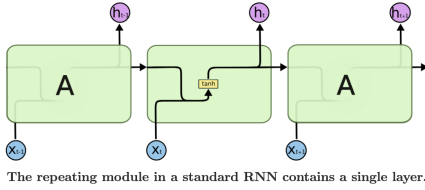


Figure 3: construction of RNN[15]

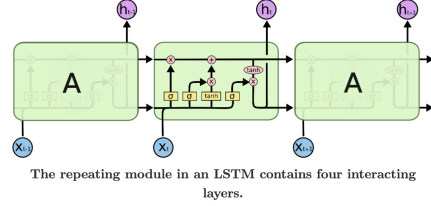


Figure 4: construction of LSTM[15]

The input gate determines how much information from the previous nerve cell layer can stay in the current memory unit, and the output gate determines how much information from the current nerve cell layer can be obtained by the next nerve cell layer. The forget gate is used to delete previously remembered but unnecessary information. [16]

4 Experiment and Discussions

4.1 ARIMA

To fit an ARIMA model, we now need to determine d , p , q for different terms.

The order of differencing: In according to the assumptions, we made, the number of times that the difference operation is performed will be determined as the d value to make the series stationary:

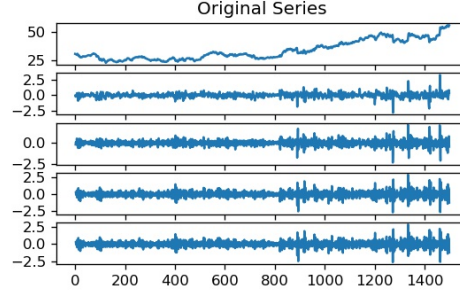


Figure 5: Stock price after differencing

After the first difference, the series ‘close price’ of ‘MSFT’ stock becomes stationary as shown in second line of Figure 5, the second and third do not make a much difference with the first, thus, we choose d of 1 which is the minimum differencing required to not over-difference the series since an over differenced series may still be stationary but in turn affects the model parameters.

The order of the AR term: The next step is to determine whether the model requires any AR terms or the length of AR terms, so we need to check partial autocorrelation, which can be regarded as the correlation between the sequence and its lag, after excluding the influence of partial lag, thus, PACF (partial autocorrelation) transmission shows a pure correlation between lag and sequence. Initially, we set the order of the AR term equal to the lags that cross the significance limit of the PACF graph shown in the Figure 6

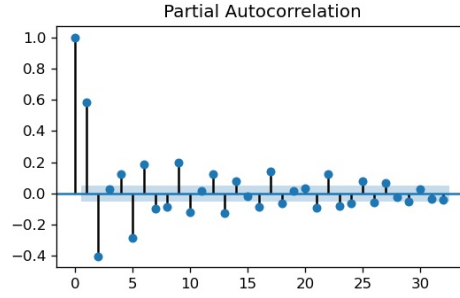


Figure 6: Partial Autocorrelation

Lag 1 and 2 are quite significant since is way higher than the significance area (blue area in the plot), though other lags are also significant but the third one is negative and the others have smaller absolute value. Thus, we fix p as 2 first and check value of one for comparison.

The order of the MA term : Similarly, we check the number of MA terms on the ACF (Autocorrelation) chart which shows the error of the lagged forecast after 1st differencing:

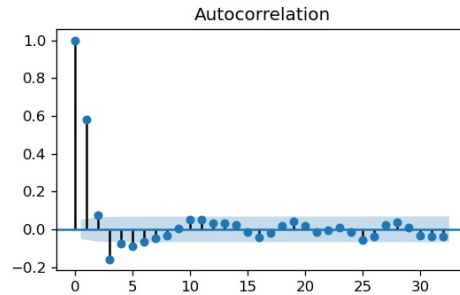


Figure 7: Autocorrelation

Two lags are much higher than the significance line and the other values, we now fix p to 2.

Therefore, we now fit an ARIMA model with order of (2,1,2) compared with other orders by AIC (Akaike information criterion) and BIC (Bayesian Information Criterion):

Table 1: AIC and BIC of models		
Order	AIC	BIC
(1, 1, 1)	490.937	512.187
(1, 1, 2)	-81.448	-54.885
(2, 1, 1)	402.135	428.698
(2, 1, 2)	-90.775	-58.900

Consistent with plots, model with order of (2, 1, 2) has lowest AIC and BIC outperforming the other models. Next, check the residuals to ensure there are no patterns:

The distribution of residual errors seems fine with mean of zero and uniform variance follows normal distribution which meets the assumption. Therefore, the model fitted well, Figure 9 also proves this, in which forecast line and true price lines basically coincide except the turning point.

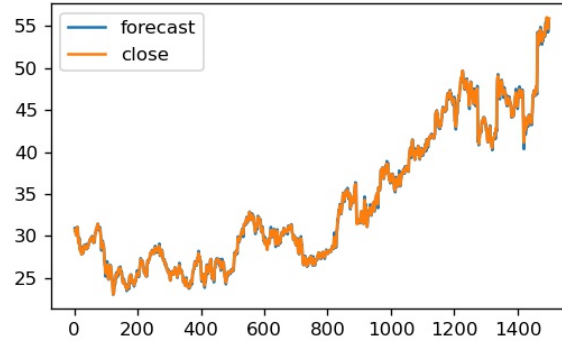


Figure 9: Fitted Model

Thus, ARIMA model may be good at predict trends, we test our model on test data to predict a long-term trend:

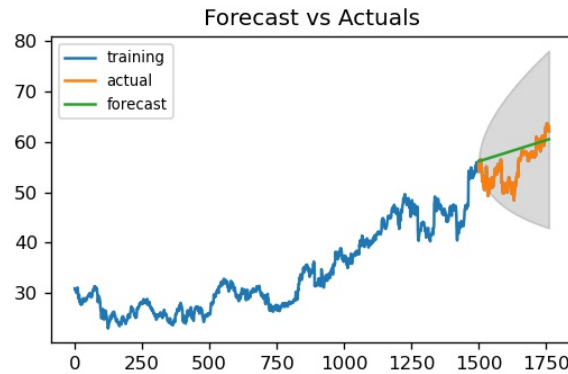


Figure 10: Long-term prediction

In Figure 10, green forecast line seems to predict a right trend in both beginning and long-term trend but it does miss the fluctuation between 1500 and 1600. Although the true close price falls in the 95 percent

confidence area which colored in grey in figure, we cannot recognize its long-term predictive ability.

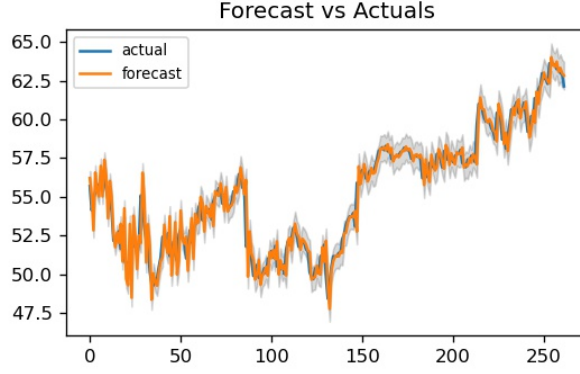


Figure 11: Short-term prediction

When looking back to short-term, we only predict one day ahead with the models fitted by past 1500 days, and similarly the grey area is 95 percent confidence area in Figure 11 the model now correctly predicts the trends with somewhat lagging though (the blue actual line is slightly on the left of forecast line), we can view the model has the ability to predict short-term trends to a certain extent. Table 2 shows evaluation metrics on test data.

Table 2: Evaluation Metrics for ARIMA regression

	Long-term	Short-term
RMSE	4.2093	0.9750
MAPE	0.0650	0.0129
ME	3.0519	-0.0051
MAE	3.4339	0.7000
MPE	0.0588	-2.4578e-06
Corr	0.7673	0.9669

ARIMA model can predict the trend of stock prices in the short-term (1 day for example), but it is invalid or meaningless in the long-term prediction and the prediction of specific prices. Although the New York Stock Exchange is the most efficient market, momentum or behavioral bias still exist, which may be related to the imbalance of information flow, but this imbalance is short or can only exist for one day (shown from ARIMA model), which is reasonable for semi-strong form markets.

4.2 Lasso

Sklearn.linear-model.Lasso from python. 10 fold is used for cross validation. α is the hyper parameter that is tuned in lasso. The best for our model is 0.0002.

To evaluate the performance of our model we calculate different metrics. Root mean square error(RMSE) is measuring the average residual error of predicted and actual values. RMSE is widely used in other stock price prediction researches and can be considered as an reliable metric to decide the robustness of the model for predicting daily prices. RMSE can be shown as follow:

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (predictedvalue - actualvalue)^2}{N}} \quad (4)$$

Where N is the number of samples. The other metric we used to evaluate our model is mean absolute percentage error(MAPE). It can be calculated as below:

$$MAPE = \frac{\sum_{i=1}^N \frac{|predictedvalue - actualvalue|}{actualvalue}}{N} \quad (5)$$

Formulas of other metrics are provided on appendix. Table 3 presents the evaluation metrics for lasso model. Figure 13 illustrates the price predicted by lasso model.

Table 3: Evaluation Metrics for lasso regression

	Train Set	Validation Set	Test Set
RMSE	0.1553	0.268	0.2699
MAPE	0.0039	0.0044	0.0039
ME	0.1175	0.2033	0.2128
MAE	0.0000	0.0415	0.0306
MPE	0.000	0.0009	0.0005
Corr	0.9996	0.9974	0.9975



Figure 12: Price prediction by Lasso

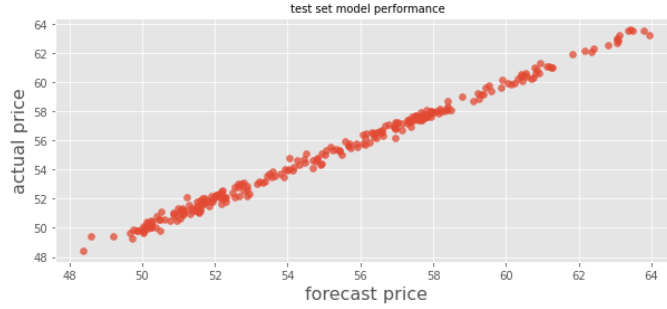


Figure 13: Actual Price versus Predicted price

RMSE and MAPE on test data set are 0.2699 and 0.0039 respectively. These small error indicates that lasso algorithm is performing well regarding to short term prediction. It can be seen in figure 13 as well.

4.3 LSTM

In this experiment, since random segmentation will destroy the chronological order, we divide the data into training set and test set in chronological order. Among them, the 2010-2015 closing price of MSFT is used as the training set, and the 2016 price is used as the test set. In terms of LSTM model construction, the data is first standardized. Since LSTM has multiple parameters that can be adjusted, in order to verify the difference in the results of different parameters, we will keep 50 neurons unchanged, and then try different neural network layers (hidden layers), data set cycle times (epoch) and single The number of data sets processed (batch_size). At the same time, use the MSE loss function and the adam stochastic gradient descent optimizer, because adam has the best overall performance and fast convergence in regression problems. Finally, we will allocate 1 neuron in the output layer to predict the normalized stock price. Below is the experimental plan.

	Neurons	Hidden layer	Batch_szie	Epoch	Droupout
Plan 1	50	2	1	1	NA
Plan 2	50	2	64	1	0.2
Plan 3	50	2	1	1	0.2
Plan 4	50	2	64	10	0.2
Plan 5	50	4	1	1	NA
Plan 6	50	4	64	1	0.2
Plan 7	50	4	1	1	0.2
Plan 8	50	4	64	10	0.2

Figure 14: Experiment Plan

By implementing different combinations, we got different results of the length of model training , Loss value and RMSE value.

	Training Time(s)	Loss Value	RMSE Value
Plan 1	50	0.0016	1.907713712
Plan 2	2	0.0176	3.351349488
Plan 3	51	0.0034	2.177552537
Plan 4	13	0.0016	1.476778609
Plan 5	97	0.003	4.130236306
Plan 6	4	0.0234	4.366130642
Plan 7	92	0.0054	2.538804984
Plan 8	35	0.0025	3.104442692

Figure 15: Experiment Plan

To see the differences more clearly, let's convert the data to charts. The following figure shows the differences in run time and loss value for different parameters.

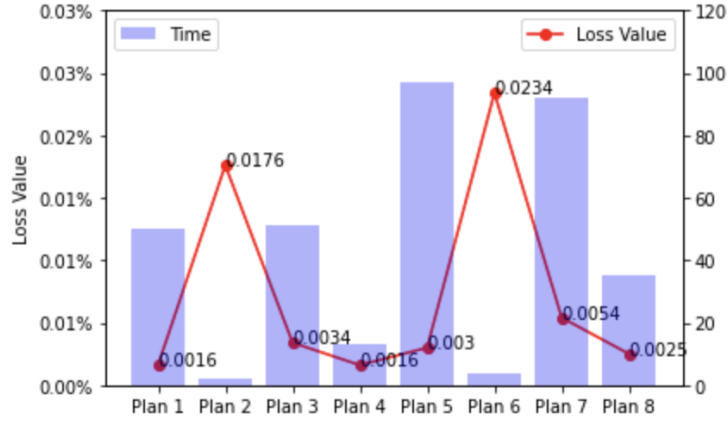


Figure 16: Running time and Loss value

From the point of view of running time and loss value, Plan 1 and Plan 4 have the lowest loss value. From Experiment Plan, Plan 1 and Plan 4 are two layers of hiding, but Plan 4's batch_size and epoch are more than plan 1, and plan 4 adds dropout values which is 0.2. As a result, we found that two layers of hidden layers perform better on this dataset than four layers of hidden layers, and are better suited to this dataset. At the same time, although the number of cycles has been increased, the run time of the model has been greatly reduced due to the increase in the batch_size and dropout values. Next we continue to look at the performance of the RMSE value of each Plan.

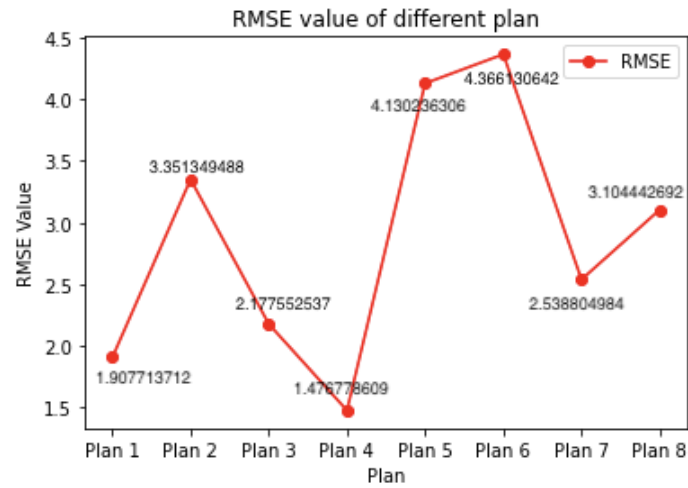


Figure 17: RMSE Value of Each Plan

We found that compared to Plan 1, Plan 4 not only takes less time, but also has the lowest loss and RMSE values. Let's take a look at the performance of plan 1 and Plan 4 in the forecast results.



Figure 18: Plan 1 Result



Figure 19: Plan 4 Result

By comparing the results, it can be seen that the forecast of Plan 4 is closer to the price trend than the forecast of Plan 1. When stock prices fluctuate, our forecasts produce the same volatility, which is basically in

line with the moving trend. For example, in 2016 there was a significant decline and rise in real stock prices, as our model predicted.

At the same time, let's look at the results of the two worst performing scenarios.



Figure 20: Plan 5 Result



Figure 21: Plan 6 Result

As can be seen from the picture above, although Plan 5 and Plan 6 performed poorly, the stock price decline and rise trend in 2016 are also forecast. So, as things stand, LSTM really seems to be a good fit for this type of problem.

4.4 Comparison

Table 4: Comparison between models

	Running Time	RMSE
ARIMA	1s	0.9750
Lasso	4s	0.2699
LSTM	13s	1.4768

We first check the running time for training model, and RMSE (Root Mean Square Error) on test set for comparison, from Table 4, ARIMA model needs least running time and Lasso model gives lowest RMSE, which means Lasso model best fits the data. In prediction from Figure 11, Figure 12 and Figure 19, all models show the abilities of predicting trends while LSTM better describe the turning points than long-term ARIMA, but Lasso gives best prediction which almost coincide with the actual price curve except few turning points, followed by short-term ARIMA. Lasso performs better may be explained by Lasso model could use multiple features, while ARIMA is limited by one piece of time series data. Therefore, Lasso regression model may be best used to describe and predict stock prices from our report.

5 Conclusion

This report uses ARIMA, Lasso, and LSTM to train Microsoft stock data, and shows in detail how to adjust hyper-parameters and predict stock price trends during the training process, and then provides an evaluation matrix and prediction images for comparison. Finally, the Lasso model gives the best prediction (the smallest error), which proves that even in the most developed stock market in the world, momentum (to follow the trends), behavioral bias and other non-information-driven phenomena still exist, but the long-term ARIMA model proves that it is not advisable to use trends to predict long-term prices, and it also reflects the efficiency of the market to a certain extent.

Although this report uses three different machine learning methods, it still does not exceed the limitations of the original theories. For example, the time series model is difficult to use a variety of data because of the requirements of auto-regression (external variables that can be added include only seasonal parameters, etc.). Therefore, a combination of multiple models may be better for forecasting. In fact, many organizations are already exploring the application of multiple models. In addition, adding external information data may be better for forecasting, because the price is essentially a response to information: using the advantages of machine learning to process big data, adding geographic information, investment-related indexes and other data are feasible improvements.

A Appendix

A.1 Instructions on codes

ARIMA: The library that needs to be loaded is in the first cell, which are parameter selection, modeling and evaluation in order.

Lasso: The code is saved as an ipynb file which can be run on jupyter notebook. First cell contains needed libraries to run the code. Some packages installation may be needed. In second cell the data file is uploaded. The data file should be downloaded and saved in the same level as jupyter notebook. For the rest of the code every cell can run individually or all together.

LSTM: In addition to the regular code library, the code running environment this time also needs to install TensorFlow and Keras.

A.2 Formula of evaluation metrics

$$ME = \frac{1}{N} \sum_{i=1}^N (\text{predictorvalue} - \text{actualvalue}) \quad (6)$$

$$MAE = \frac{1}{N} \sum_{i=1}^N |\text{predictorvalue} - \text{actualvalue}| \quad (7)$$

$$MPE = \frac{1}{N} \sum_{i=1}^N \frac{(\text{predictorvalue} - \text{actualvalue})}{\text{actualvalue}} \quad (8)$$

Where N is the number of samples.

Corr is correlation coefficient of prediction value and actual value

References

- [1] Engle RF. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the Econometric Society*. 1982;987–1007.
- [2] Bollerslev T. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*. 1986;31(3):307–327.
- [3] Morgan J. Riskmetrics—Technical Document. New York: JP Morgan and Reuters; 1996.
- [4] Merh N, Saxena VP, Pardasani KR. A comparison between hybrid approaches of ANN and ARIMA for Indian stock trend forecasting. *Business Intelligence Journal*. 2010;3(2):23–43.
- [5] Roy SS, Mittal D, Basu A, Abraham A. Stock Market Forecasting Using LASSO Linear Regression Model. In: Abraham A, Krömer P, Snasel V, editors. *Afro-European Conference for Industrial Advancement*. Cham: Springer International Publishing; 2015. p. 371–381.
- [6] Setiawan B. LASSO Technique Application in Stock Market Modelling: An Empirical Evidence in Indonesia; 2018. .
- [7] Pathak A, Shetty NP. Indian Stock Market Prediction Using Machine Learning and Sentiment Analysis. In: Behera HS, Nayak J, Naik B, Abraham A, editors. *Computational Intelligence in Data Mining*. Singapore: Springer Singapore; 2019. p. 595–603.
- [8] Ticknor JL. A Bayesian regularized artificial neural network for stock market forecasting. *Expert Systems with Applications*. 2013;40(14):5501 – 5506. Available from: <http://www.sciencedirect.com/science/article/pii/S0957417413002509>.
- [9] Chen Y, Yang B, Abraham A. Flexible neural trees ensemble for stock index modeling. *Neurocomputing*. 2007;70(4):697 – 703. *Advanced Neurocomputing Theory and Methodology*. Available from: <http://www.sciencedirect.com/science/article/pii/S0925231206002736>.
- [10] Gers FA, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM. 1999.
- [11] Chen L, Wang Z, Wang G. Application of LSTM network in short-term power load forecasting under the framework of deep learning. *Electric Power Information and Communication Technology*. 2017;(5):8–11.
- [12] Sundermeyer M, Schlüter R, Ney H. LSTM neural networks for language modeling. In: *Thirteenth annual conference of the international speech communication association*; 2012. .
- [13] Efron B, Hastie T, Johnstone I, Tibshirani R. Least angle regression. *Ann Statist*. 2004 04;32(2):407–499. Available from: <https://doi.org/10.1214/009053604000000067>.
- [14] Tibshirani R. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society Series B (Methodological)*. 1996;58(1):267–288. Available from: <http://www.jstor.org/stable/2346178>.
- [15] Understanding LSTM Networks; 2015. <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>.
- [16] Hochreiter S, Schmidhuber J. Long short-term memory. *Neural computation*. 1997;9(8):1735–1780.