The University of Hong Kong
Department of Computer Science
COMP7705 MSc(CompSc) Projects

Financial Feature Analysis and Prediction

Interim Report

Members: Chen Zixuan (3035658545)
Chen Pengxiu (3035668435)
Peng Wanlan (3035657943)
Yan Jun (3035658296)
Ye Siqi (3035657450)

Supervisor: Professor S.M.Yiu

7th June 2020

Contents

Abstract

After 21st century's coming, large amounts of traditional industries have changed a lot as computer science developed. And one of the most shining stars in computer science must be Artificial Intelligence (AI). As students who major in computers science, we would like to make use of AI in traditional financial industry.

As a kind of financial data, stock prices have lots of features and they are not that similar to other popular topics in AI like computer vision and natural language processing. Stock prices usually have abstract features and vague connections. Amounts of traditional methods to do financial computing are always based on mathematical formulas and functions. To analyse stock prices, we usually need to do lots of mathematical derivation with trying parameters and models repeatedly.

However, with the help of machine learning and feature engineering, we want to speed up this procedure. There have been lots of models and methods in this area nowadays. Our goal is to improve the existing models and hope to do some creations to seek the chance of getting a new model.

Acknowledgement

We would like to express our great expression to the members in our group. All the members in our group work hard and made lots of efforts to realize our targets. And we have to thank our supervisor Professor S.M.Yiu specially for his assistance. His suggestions and generosity helps us a lot to reach our current stage.

List of Figures and Tables

Abbreviations

AI – Artificial Intelligence

HSI - Hang Seng Index

AR – Auto Regression

MA - Moving Average

MACD - Moving Average Convergence/Divergence

ARMA - Autoregressive moving average model

ARIMA - Autoregressive Integrated Moving Average Model

RSI - Relative Strength Index

RNN - Recursive Neural Networks

LSTM - Long Short-Term Memory

SVM - Support Vector Machine

CNN - Convolutional Neural Network

MLP - Multi-Layer Perceptions

1. Introduction

The following part will briefly introduce our initial idea and our scope of this project. First of all, we will describe the background of our project. Secondly, we will introduce some existing and possible methods. Thirdly, we hope to show our objective and scope. At last, we will show the structure of this report.

1.1 Background

Generally, people do stock price prediction with time series analysis. In the procedure of time series analysis, we always use several kinds of Moving Average (MA), variance and standard deviation to grasp the features of the price. Then, we can apply certain index to chase the trend, such as: Moving Average Convergence/Divergence (MACD), Relative Strength Index (RSI) and Boll Line. These methods can really help us traditionally.

However, with the development of Artificial Intelligence, we want to apply the methods of deep learning in stock price analysis.

In the first step, we use historical data only. After the pre-processing of data, we want to use RNN neural network and add LSTM layer to it. Thus, after training, we can verify the output and compare the results with traditional methods.

In the second step, we want to add more message and features in the data like real-time public opinions and compare the result with the former work. In the third step, we would like to try some other networks and add certain layer to optimize the results.

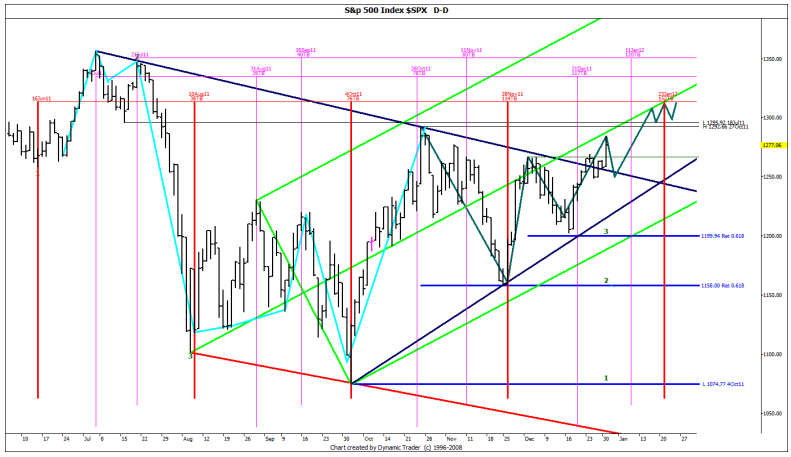


Figure 1

1.2 Existing Method

Stock market data are highly complex and difficult to predict, even for human experts, due to a number of external factors, e.g., politics, global economy, and trader expectation. The trends in stock market data tend to be nonlinear, uncertain, and non-stationary. In most of the cases, proposed approaches have been mainly based on statistical models used to analyse time series of interest. such as: Moving Average Convergence/Divergence (MACD), Relative Strength Index (RSI) and Boll Line.

However, the problem of time-series is that stock prediction is often viewed as a single channel problem, which explains the difficulties to produce accurate prediction systems, since stocks depend on a myriad of other factors, and arguably not at all on past values of the stock itself.

On the other hand, with the development of deep learning, this area has aroused the interest of academia and practitioners, especially in the field of finance. Deep learning methods offer better representation and classification on a multitude of time-series problems compared to shallow approaches when configured and trained properly. As a result, different types of models have been considered about solving the problem of predicting asset price movements and the behaviours in time of more structured financial instruments. In previous works, better predicting results have been obtained using linear classifiers as the logistic regression one, which has been used to predict the Indian Stock market[1], or with respect to the S&P500 index[2]. More complicated techniques, as large-margin classifier or Support Vector Machine (SVM) were also been tried.

After the rise of neural network. Based on the type of application, various types of deep neural network architectures are used in financial areas. These include multi-layer perceptions (MLP), Recursive Neural Networks (RNN), Long Short-Term Memory (LSTM), CNN (Convolutional Neural Network) etc[3]. A recurrent neural network (RRN) is any artificial neural network whose neurons send feedback signals to each other. The idea behind RNNs is to make use of sequential information. In a traditional neural network, we assume that all inputs (and outputs) are independent of each other. But for many tasks this is not the better idea. In particular, if one wants to predict the next word in a sentence, then it is better if he knows which words came before it. RNNs

are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations. With the introduction of LSTM[4], the analysis of time dependent data become more efficient. These types of networks have the capability of holding past information and they have been used in stock price predictions[5][6].

However, most approaches of RNN applied to stock prediction have given unsatisfactory results. This is because the price of a stock is not only related to its own historical trend, but also to a large extent will be affected by emergencies and public opinions. A major earthquake in Chile, for example, would shake public opinion and would inevitably affect the share prices of Chile's proprietary energy.

In summary, it can be concluded that there is still room to improve existing techniques for making safe and accurate stock prediction systems. If additional information from sources that affect the stock market can be measured and obtained, such as general public opinions from social media[7], trading volume[8], market specific domain knowledge, and political and economic factors, it can be combined together with the stock price data to achieve higher stock price predictions[9].

1.3 Objective and Scope

In this project, we aim to analyse the price of several stocks in Hang Seng Index (HSI) in the past decade and seek the features to help us predict its trend. We hope to optimize the existing methods to improve the predicting results. We also want to improve the procedure of prediction and try to predict the index or portfolios not only single stocks. Although stock price prediction is not a novel topic, we still want to do something different in this field.

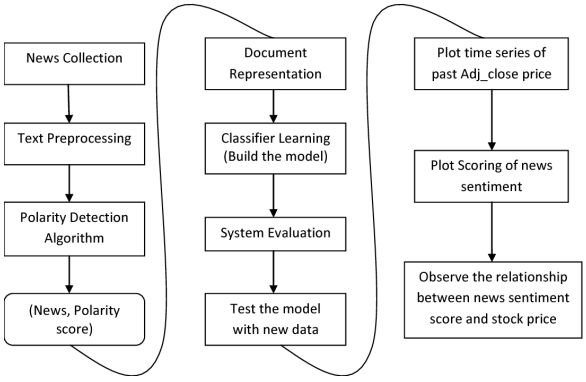


Figure 2

1.4 Report Outline

The structure of this report is as followings. Chapter 2 will introduce the methodologies we used currently. Chapter 3 will describe the data processing and training results in our experiment. Chapter 4 will contain our current status and findings. Chapter 5 will be a brief conclusion and our future work.

2. Methodology

We try several different methods to seek a better result in the prediction of stock prices. Both traditional method and machine learning methods are tried. We will choose from the methods with better effects and find possible optimizations in the future work.

2.1 ARIMA

In general statistic models, time series is really useful. Time series means to construct an array of the same statistic index according to their time and predict the future data with the help of historical data. Common time series have 4 parts: Auto-Regression (AR), Moving Average (MA), ARMA (Autoregressive moving average model), ARIMA (Autoregressive Integrated Moving Average Model).

We have to describe the main conditions and formulas we use in this part. Firstly, if time series y_t satisfies that:

$$Y_t = \mu + \sum_{i=1}^{p} \beta_i Y_{t-i} + \epsilon_t$$

ϵ_t is an individual series with same distribution. And it satisfies:

$$VAR\epsilon_t = \sigma_\epsilon^2 > 0, E(\epsilon_t) = 0$$

then, we can say y_t satisfies the p-order AR model.

Secondly, if time series y_t satisfies that:

$$Y_t = \mu + \epsilon_t + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}$$

then, we call y_t satisfies the p-order MA model.

Thirdly, if time series y_t satisfies that:

$$Y_t = \mu + \sum_{i=1}^{p} \beta_i Y_{t-i} + \epsilon_t + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}$$

then, we call y_t satisfies the (p, q)-order ARMA model.

At last, after process of difference, ARMA model will become ARIMA model.

2.2 XGBoost

The XGBoost algorithm improves the Gradient Boosting Decision Tree (GDBT) algorithm. It realizes the generation of weak learners by adding the regular term in the loss function to reduce the risk of overfitting, Besides, the XGBoost algorithm use the first and second derivative values of the loss function instead of the linear search method. Other techniques such as preordering, weighted quantiles, sparse matrix recognition, and cache recognition greatly improve the performance of the XGBoost algorithm as well.

The data set is described as $D\{(x_i, y_i)\}(|D| = n, x_i \in R^m, y_i \in R)$, which has n samples and m features. The CART space is described as $F = \{f(x) = w_{q(x)}\}(q: R^m \to T, w \in R^T)$, in which q is the structure leaf node coefficient, T is the number of leaves, each f_k corresponds to an independent tree structure q and leaf weights w. The XGBoost algorithm minimizes the following regularized objective function:

$$L(\emptyset) = \sum_i l(\hat{y}, y_i) + \sum_k \Omega(f_k)$$

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2$$

where l is the loss function, Ω is the penalty term for the complexity of the model, γ is the L1 regularization coefficient, and λ is the L2 regularization coefficient.

XGBoost constructs a compressed data block before constructing the CART tree. It stores a pointer to the sample index after sorting by each feature value. Each time it searches for the optimal split point, it only needs to traverse in this data block once, and then according to its pointer to extract the first and second derivatives of each sample, which greatly improves the calculation speed.

XGBoost distributes data to multiple available disks and sets up a data loading thread for each disk, so that data can be loaded from multiple disks into the memory buffer at the same time. Moreover, the model training thread can read data from each buffer freely, which greatly increases data throughput and makes full use of the limited resources of the computer.

2.3 LSTM

2.3.1 Structure and Principle of LSTM

Long short term memory (LSTM) is a kind of time cycle neural network, which is specially designed to solve the long-term dependence problem of general RNN. All RNN have a chain form of repetitive neural network module. LSTM is different from the standard RNN network in that there is only one network layer in the unit, and there are four network layers inside it. It adopts the mechanism of control gate, which is composed of memory cell, input gate, output gate and forgetting gate. The specific structure is shown in Figure 1 and Figure 2.

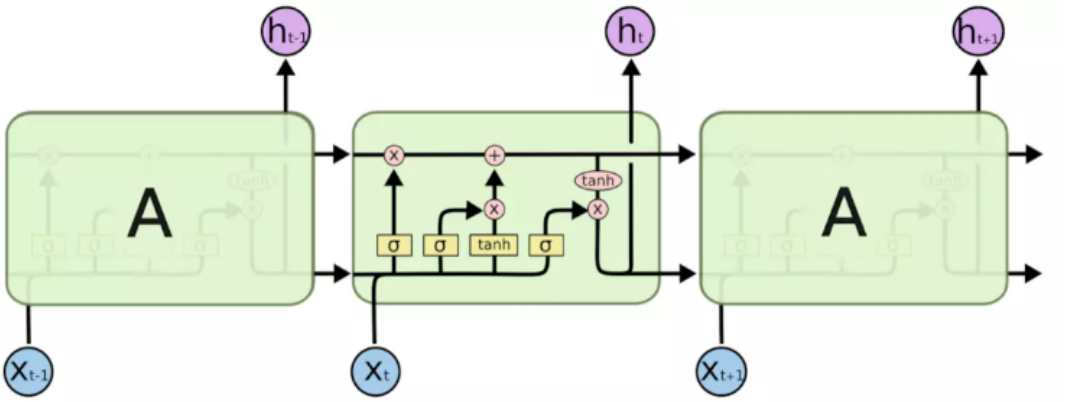


Figure 3

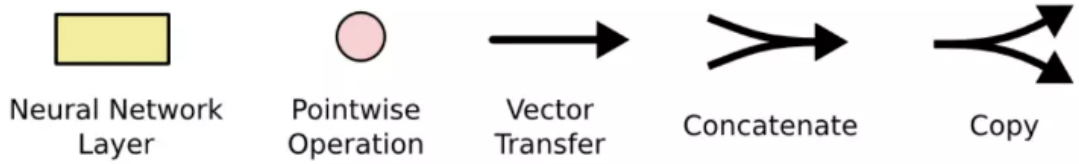
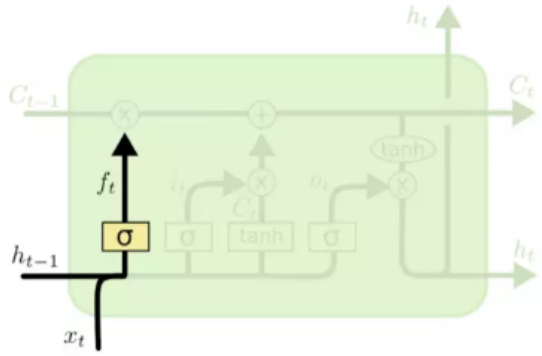


Figure 4

In the structure diagram, x_t represents the input of time t, h_t represents the state value of cells at time t. Thus the three different big boxes represent the state of cells in different time sequence. For the four neural network layers in yellow, there are 3 layers with sigmoid activation function and one layer with tanh activation function. The number of hidden neurons in the feedforward network layer needs to be debugged and trained to get the best value.

The first step of LSTM is to determine what information the cell state needs to discard. This part of the operation is handled by a sigmoid unit called forgetting gate. It outputs a vector between 0-1 by viewing and information. The value of 0-1 in the

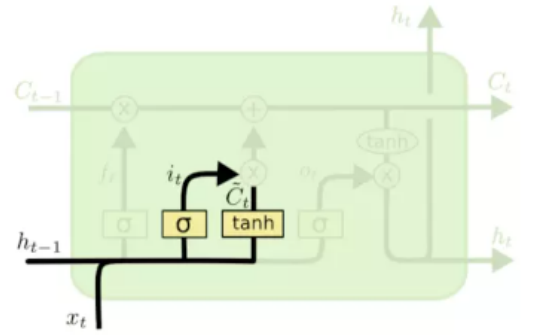
vector indicates which information in the cell state is reserved or discarded. 0 means no reservation, 1 means all reservation. Forgetting door as shown below.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 5

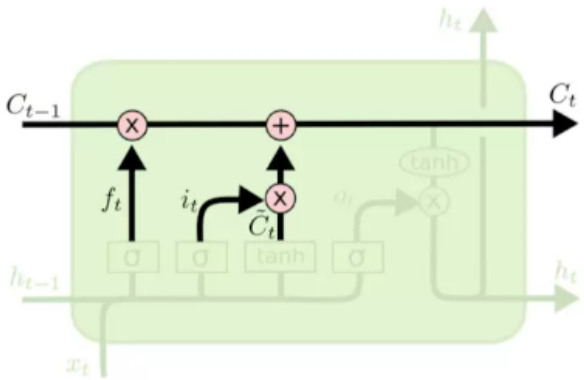
The next step is to decide what new information to add to the cell's state. This step is divided into two steps. First of all, use h_{t-1} and x_t to decide which information to update through an operation called input gate. Then we use h_{t-1} and x_t through a tanh layer to get new candidate cell information $\tilde{C}_t$, which may be updated to cell information. These two steps are described in the following figure.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 6

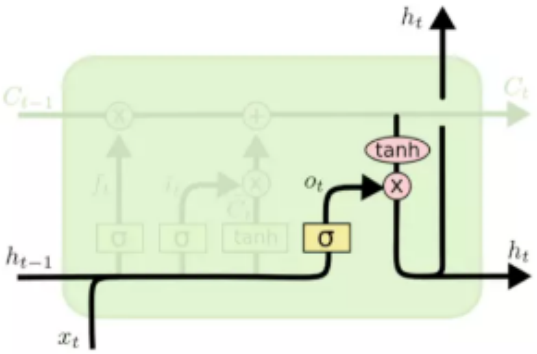
Next, update old cell information C_{t-1} becomes a new cell information C_t. The updated rule is to choose to forget part of the old cell information by forgetting gate and add the part of candidate cell information $\tilde{C}_t$ by input gate to get new cell information C_t.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 7

After updating the cell state, we need to determine which state characteristics of the output cell are based on the input h_{t-1} and x_t. Here, we need to pass the input through a sigmoid layer called the output gate to get the judgment conditions, and then pass the cell state through the tanh layer to get a vector of values between - 1 and 1, which is multiplied by the judgment conditions obtained by the output gate to get the final output of the RNN unit.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 8

2.3.2 The Construction of LSTM Model

This experiment uses the Keras framework. There are 2 models of Keras: Sequential and Model. In this experiment, Sequential is used to build two-layer LSTM network model to analyse and predict the same stock. For the case that the data scale of stock is not very large, the experiment uses LSTM network and full connection mode to predict the short-term and long-term price of stock. The prediction performance evaluation of the model refers to using root mean square error (RMSE) and model prediction accuracy (accuracy) to compare the experimental results. The figure below shows part of the experimental code of the two-layer LSTM and full connection layer network model.

3. Data Processing and Training

3.1 Data Processing

The data sets used in this experiment contains the Hang Seng Index and the stock data of 50 constituent companies from January 4, 2020, to March 5, 2020. In particular, the data for training LSTM model is mainly the opening price and closing price of individual companies from January 4, 2010, to March 5, 2020.

The original data obtained may be disordered and lack of value, so interpolation, sorting and other operations are needed, so as to get regular stock time series data, and further build a complete and effective data set.

What's more, as the value ranges of various companies' stock prices are inconsistent, the ranges may cause an unsteady effect on the training. It is necessary to adopt normalization to reduce the impact of the range on training. We use Min-Max scaling to scale the original data to the range of [0, 1]. For the opening price, closing price, highest price, and lowest price calculate their maximum and minimum values, respectively, and then perform the normalization operation.

3.2 Data Training

When optimizing the machine learning model parameters, it is essential to choose the appropriate cross-validation method for parameter tuning.

In this project, we choose a cross-validation method that guarantees the data utilization rate and preserves the relationship between the time series data, which is the time series cross-validation. In time series cross-validation, the training set and the verification set will follow the principle of time sequence during the division process. The end time of the training set is the start time of the verification set. It can use past data to train the model and use future data to verify to avoid using future rules to predict the historical results of "cheating" behaviour.

Taking the 5-fold sequential cross-validation method as an example:
In 5-fold time-series cross-validation, the sample must first be divided into five parts according to the time sequence. The first training process uses the first part as the

training set and the second part as the validation set; the second training process uses the third part as the validation set, the first two parts as the training set. There are four verification processes in the sample, and finally, the average performance of the model on the four verification sets.

And in the parameter optimization of XGBoost, we have so many hyperparameters, while not all parameters will have a significant impact on the model's prediction effect. Changes in specific parameters will cause the model's prediction effect to show a random change or unchanged state.

There are three main parameters that can significantly affect the model. The parameters and their interpretations are shown in the following table.

Parameter Name	Meaning
learning rate	The weight reduction coefficient v of each weak learner is also called step size, and the value range of v is 0<v<1. For the same training set fitting effect, the smaller v means the more iterations of the weak learner.
max_depth	The maximum depth of the decision tree. The larger the value, the more complicated the model, and the more likely it is to overfit.
subsample	The value range is (0,1]. 1 means using all samples. If the value is less than 1, it means that only a part of the samples participating in the fitting process.

This paper uses the most extensive grid search method in the field of machine learning to optimize the above three main parameters within a given range. Then, select the hyperparameters set with the highest average accuracy in the verification set as the final hyperparameters of the model. In the end, we find that the optimal values of the hyperparameters are learning rate (0.01), max_depth (50), and subsample (0.9).

4. Results and Findings

In this project, Since stock price prediction may not be that precise, we take a threshold of 0.01% of the correct result to judge whether the predicted result is correct or not. Under this assumption, we tried three methods to compare the effects and seek for possible optimization in the future work. What's more, in current stage we only predict price of single stocks. We will put our attention on index and portfolios in the future.

4.1 ARIMA

The prediction of stock price is as following (take Tencent as an example):

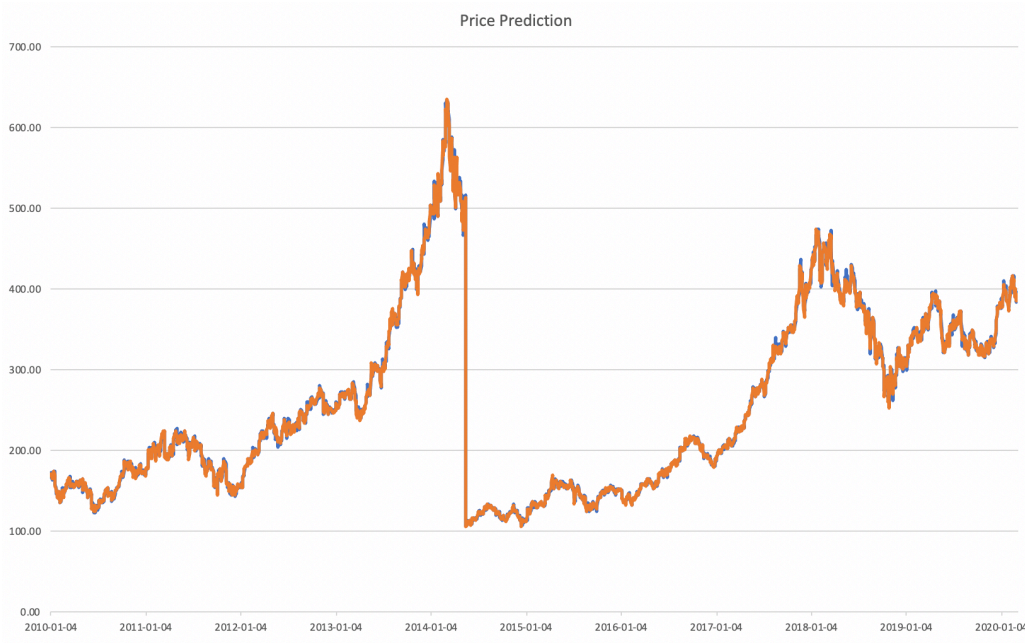


Figure 9

And the rate of correctness is 84.17%.

4.2 XGBoost

The following figure shows the real stock price and predicted values. Blue line for actual price and red line for prediction. The first picture contains data for the entire statistical period, while the second one displays the test part only.

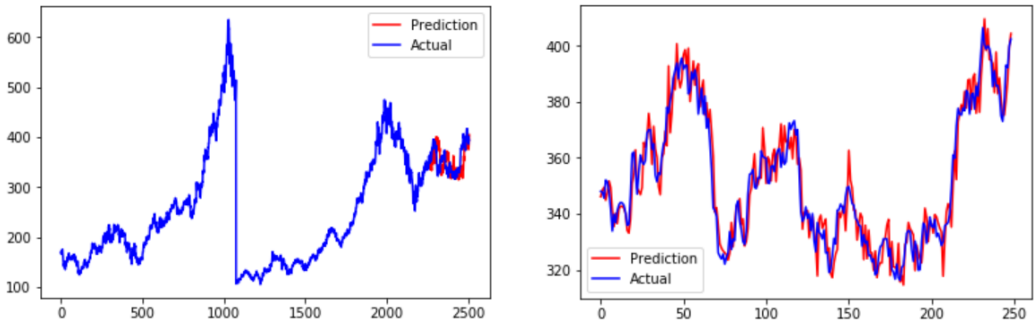


Figure 10

The curve for prediction always has unexpected peaks and troughs, but the overall trend is basically the same as the actual price. The predicted value has obvious lag to the price change, and it is difficult to accurately identify important stock price change points. The accuracy of prediction is 89.61%.

4.3 LSTM

In this algorithm, we take the test part to show the result.

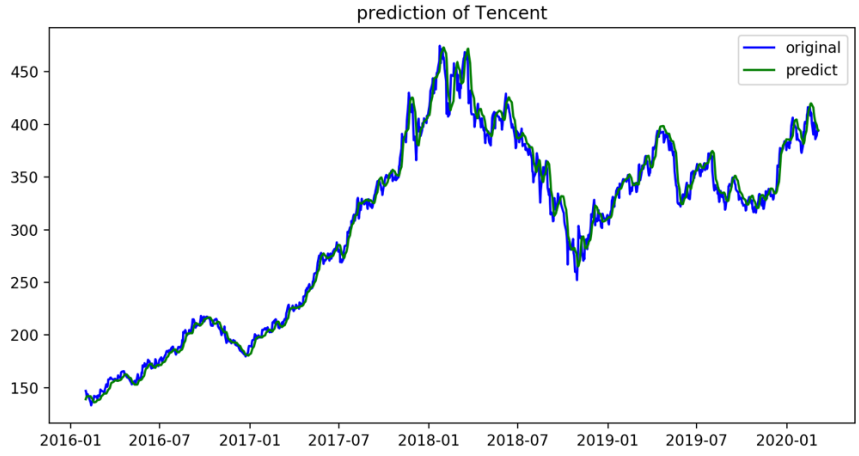


Figure 11

And the rate of accuracy is 87.70%.

5. Conclusion and Future Work

According to the result above, we can find there still some space for optimization and we have to get more information and more algorithms to get better results. For example, ARIMA has the requirement of stationarity of data. However, stocks always have certain unpredictable factors which can influence the price greatly. In this case, predicting a single stock becomes challenging and unreliable. Compared to single stocks, portfolios and indices are more reliable and less risky. Then, predicting portfolios and indices can also be a possible way to get more precise price prediction.

In the future work, we aim to optimize the existing algorithms we used and find certain rules to construct portfolios to reach better results and generality. Our goals in the future are as following:

	Tasks	***Estimated completion time***
1	Project Webpage	June 15, 2020
2	Optimization of machine learning methods	July 1, 2020
3	Poster design	July 15, 2020
4	Final report and conclusion	August 1, 2020

References

[1] S.-H.Chen, Genetic Algorithms and Genetic Programming in Computational Finance, Springer, 2002

[2] A. Dutta, G. Bandopadhyay and S. Sengupta, Prediction of Stock Performance in the Indian Stock Market Using Logistic Regression, Inter- national Journal of Business and Information, 2012

[3] J. Heaton, N. Polson, and J. Witte, "Deep learning in finance," arXiv preprint arXiv:1602.06561, 2016.

[4] H. Jia, "Investigation into the effectiveness of long short term memory networks for stock price prediction," arXiv preprint arXiv:1603.07893, 2016.

[5] Y. Bengio, I. J. Goodfellow, and A. Courville, "Deep learning," Nature, vol. 521, pp. 436–444, 2015.

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

[7]Bollen, J., Mao, H., Zeng, X., 2011. Twitter mood predicts the stock market. Journal of Computational Science 2, 1-8

[8]Zhu, X., Wang, H., Xu, L., Li, H., 2008. Predicting stock index increments by neural networks: The role of trading volume under different horizons. Expert Systems with Applications 34, 3043-3054.

[9]Agrawal, J.G., Chourasia, V.S., Mittra, A.K., 2013. State-of-the-art in stock prediction techniques. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 2, 1360-1366.