


Financial Series Prediction: Comparison Between Precision of Time Series Models and Machine Learning Methods

David Jiang

Related papers

[Download a PDF Pack](#) of the best related papers 



[Reliability of ARMA and GARCH models of electricity spot market prices](#)

Piotr Ptak

[STOCK PRICE PREDICTION USING LSTM, RNN AND CNN-SLIDING WINDOW MODEL](#)

Kai Shun

[Electricity price forecasting: A review of the state-of-the-art with a look into the future](#)

Rafał Weron

Financial Series Prediction: Comparison Between Precision of Time Series Models and Machine Learning Methods

Xin-Yao Qian

December 1, 2017

Abstract

Investors collect information from trading market and make investing decision based on collected information, i.e. belief of future trend of security's price. Therefore, several mainstream trend analysis methodology come into being and develop gradually. However, precise trend predicting has long been a difficult problem because of overwhelming market information. Although traditional time series models like ARIMA and GARCH have been researched and proved to be effective in predicting, their performances are still far from satisfying. Machine learning, as an emerging research field in recent years, has brought about many incredible improvements in tasks such as regressing and classifying, and it's also promising to exploit the methodology in financial time series predicting. In this paper, the predicting precision of financial time series between traditional time series models ARIMA, and mainstream machine learning models including logistic regression, multiple-layer perceptron, support vector machine along with deep learning model denoising auto-encoder are compared through experiment on real data sets composed of three stock index data including Dow 30, S&P 500 and Nasdaq. The result shows that machine learning as a modern method actually far surpasses traditional models in precision.

Keywords: Security trading, time series, machine learning, deep learning

1 Introduction

Financial time series prediction is an interesting topic that has been researched broadly. The motivation to research on it lies at investor's investing reference: more precise prediction results can effectively assist investors in deciding profitable investment strategies.

In financial series prediction, one popular problem is to predict security's future price trend, because a investor forms personal belief of security's future price trend, and make investing decision based on the belief. This paper also focused on trend predicting problem.

Among traditional predicting method, there are several classical models which have proved to be useful theoretically. For example, Box-Jenkins ARIMA makes use of information underlying in lag terms of variable itself and errors in the past; GARCH can capture variability of variance effectively to help judge volatility of stock return. Also, there are many other derivative models like nonlinear GARCH(NGARCH), integrated GARCH(IGARCH), exponential GARCH(EGARCH), which can perform well in situations with different settings.

With the fast development of computing and storing ability of computers, *machine learning (ML)* has been an emerging study field of computer science and statistics, and has brought many great enhancements in tasks concerning predicting within all kinds of fields. ML mainly takes use of artificial algorithms (e.g. artificial neural network) to learn *pattern*, underlying given data, and derive a model with ability to predict on new data with the same structure. ML shows outstanding performance in predicting using algorithms most of which have been proved effective rigidly in mathematics. So, it's also quite exciting and promising if ML can be exploited in financial time series predicting.

Also, *deep learning (DL)* has become a increasingly popular subfield of ML in recent years because of its surprisingly outstanding performance on precision in many fields such as computer vision, automatic speech recognition, natural language processing and so on . It mainly consists of various derivatives of *artificial neural network*, which includes models like convolutional neural network (CNN), recurrent neural network (RNN), deep belief network (DBF) and so on. Therefore, we will also introduce some applicable DL models in later comparison in predicting precision.

Later sections of this paper mainly consists of four parts:

- (i) Review of previous research work on financial time series predicting
- (ii) Brief introduction of principles of traditional time series models and machine learning models including deep learning models, all of which are to be compared in later prediction experiment.
- (iii) Experiment of predicting on real financial data using different models and comparison between performance of different models.
- (iv) Conclusion and advice for further research.

2 Literature Review

Financial time series forecasting has long been a topic of interest. Traditional mainstream methods to deal with the problem mainly consist of fundamental analysis and technical analysis, while there has been more and more trials with introducing new edge-cutting methods, i.e. machine learning into prediction in recent years.

2.1 Fundamental Analysis

In general, fundamental analysis tries to analyze some macro features a company or business is showing. It based its principles that the market value of a stock tends to move towards its “real value” or “intrinsic value” [4]. For investors who are in favor of fundamental analysis, he/she would collect information that can help assess the prospect of goal company or business, and if he came to the conclusion that the current value of certain company is underestimated, then he would expect the stock value of the company to return to its higher “true value” later. Therefore, he could buy stocks now and got premium in the future. Or if he concluded that the company is overestimated, then he could short sell the stocks. There are some widely used indicators that can help assess a company’s true value, such as *the price-to-book ratio*, *price-to-earnings ratio* and so on, which are calculated using public information of the company.

However, fundamental analysis is limited in that it’s useful for trend predicting, i.e. it’s better to exploit it in assisting to design long-period trading strategy instead of short-period one.

2.2 Technical Analysis

Technical Analysis is an analysis methodology for forecasting the direction of prices through the study of past market data, primarily price and volume [6]. Generally, investors using this methodology would formulate their trading strategy based on some technical indicators calculated by price, volume and time. Various indicators are results of different formula, and investors establishes their own beliefs of the patterns underlying the relationships between the indicators and stock price’s movement direction.

2.3 Analysis Based on Models

Time series models and machine learning models are essentially independent of technical analysis and fundamental analysis. They based on mathematical theories, and derive useful models through inputing training data. Derived model can then be used to predict on new data.

Kim made a incipient contribution to the introduction of machine learning models into financial predicting [5], and specifically, he uses Support Vector Machine (SVM) to predict the movement direction of stock price with precision measured by hit ratio (57.831%), and it outperforms Back-Propagation Network (BP Network) (54.7332%) and case-based (51.9793%) reasoning. S.L. Ho compares ARIMA with two kinds of BP Network and Recurrent Neural Network (RNN), and it turns out that both ARIMA and RNN perform well on short-period prediction than BP Network [11], while Iqbal concludes that artificial neural network (ANN) is still the best tool to predict financial time series after that he compares the performance of ARIMA with various ANN models [14]. Hossain compares GARCH(1,1) with SVM and Neural Network on the precision of volatility prediction on six stock markets indices, and it turns out that GARCH(1,1) and SVM are better on the whole than BP Network [1].

Deep learning (DL) thrives in recent years as one subfield of machine learning, which mainly develop based on artificial neural network with complex framework. DL becomes increasingly popular because its amazingly powerful performance, mainly because of the fast advancement of computing power of computer and better

algorithms for trading deep neural network. DL has been proved to surpass lots of traditional methods used in AI fields like speech recognition, visual object recognition and object detection [7]. For application in financial trend prediction, Troiano shows that both auto-encoders (AE) and restricted boltzmann machines (RBM) are effective in feature reduction for later financial series' trend prediction by SVM, while it turns out that AE can be trained faster and is more accurate than RBM [13]. Heaton illustrates that AE and Long Short Term Memory Models (LSTMs) can be put into use for many Financial problems such as Factor Models, Default Detection and Event Study [3].

3 Methodology

3.1 Model Introduction

3.1.1 ARIMA

ARIMA is a generalization model of ARMA which is a combination of AR process and MA process, and ARIMA is capable of dealing with time series data with feature of non-stationary because of its “integrate” step, which is denoted by “I” in “ARIMA”. If a time series is stationary, then we can draw ACF and PACF figure of the series data to help decide the order of ARIMA model. ARIMA’s common form can be written as below:

$$y_t = \sum_{i=1}^p \alpha_i y_{t-i} + \sum_{i=1}^q \beta_i u_{t-i} + u_t.$$

y_t stands for our goal variable to forecast, and we would fit the above model to the time series data to obtain fitted model, using which we can forecast value of y_{t+1} , y_{t+2} and so on. p and q here denote the orders of AR process and MA process respectively.

3.1.2 LR (Logistic Regression)

The principle of LR is simple though, it’s a classification method quite popular in many realistic applications owing to its good performance. The basic model can be written as:

$$y = \frac{1}{1 + e^{-\beta^T Z}} + u_t$$

where:

$$\beta = (\beta_0, \beta_1, \beta_2, \beta_3, \dots, \beta_n)^T$$

denotes parameter vector, and

$$Z = (x_0, x_1, x_2, x_3, \dots, x_n)^T$$

denotes the feature vector. The function $y = \frac{1}{1+e^{-x}}$ in fact represents a distribution function called *logistic distribution*, whose figure is depicted in Figure 1.

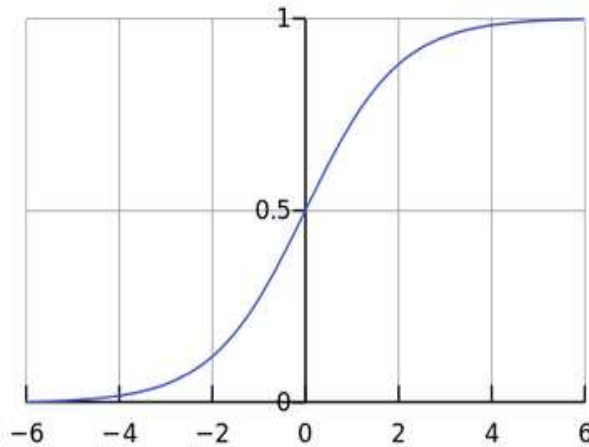


Figure 1: Logistic Curve

LR obeys such classification rule that:

$$classification = \begin{cases} 1 & \text{if } y > 0.5 \\ -1 & \text{else} \end{cases}$$

We fit the logistic function using our training data, which consists of quantities of feature vectors Z_i to obtain fitted parameters β and then use the fitted model to do predicting job. The fitting process normally uses *maximum likelihood estimation*.

3.1.3 SVM (Support Vector Machine)

Vapnik invented SVM and improve the model by introducing kernel method and soft margin later [2]. For sample points which belong to only two classes denoted by feature vectors in R^2 space, SVM tries to find a best straight line that can rightly classify most of data points. As is shown in Figure 2, all L_1, L_2 and L_3 can divide two classes of data points correctly, but L_2 tend to be the “best” as it helps to achieve maximum distances from both nearest data points to L_3 , the classification line w.r.t. two classes, which would equip the line with strong ability of generalization, i.e. strong power to classify other data points not given in training set correctly. In general, SVM can be extended to high-dimensional space(R^n), within which data points also become high-dimensional feature vectors, while SVM would try to find a *hyperplane* to do the similar job illustrated above.

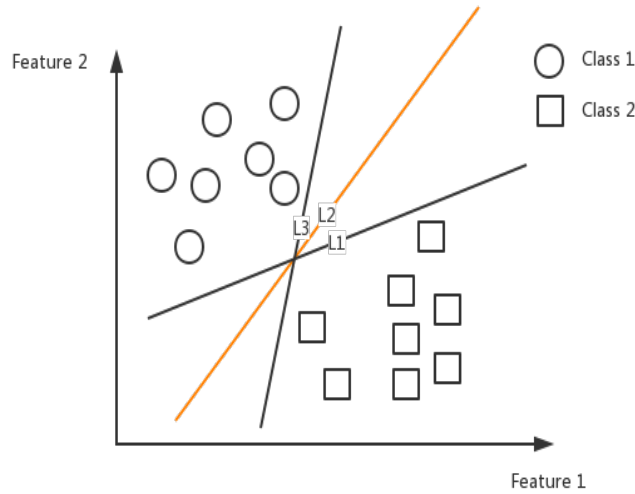


Figure 2: SVM's division of data points

Kernel method used in SVM allows it to deal with classification job concerning non-linear relation between class and feature vector of sample, which makes SVM a powerful tool in non-linear relation discovering. Besides, different from many traditional ML models who try to minimize *empirical risk*, i.e. the training error on training set, SVM tried to minimize *structural risk*, a combination of both Empirical Risk and *complexity* of model, which effectively avoids the problem of overfitting, which means that fitted model can perform well on training set but bad on test set.

3.1.4 MLP (Multilayer Perceptron)

MLP is a kind of *artificial neural network*, a series of model partly inspired by working principles of human brain. It fits a non-linear function(mapping) from input variable vector to output value vector. It's essentially a directed graph consists of layers of *neurons* linked with each other. Every link between neurons has a weight. As is shown in Figure 3, every circle stands for a neuron, while a line of neurons forms a *layer*. Layers in MLP can be mainly classified into input layer, hidden layer and output layer according to their different role. Input layer accepts input variable vector, and output layer outputs the result of processing input vector. As

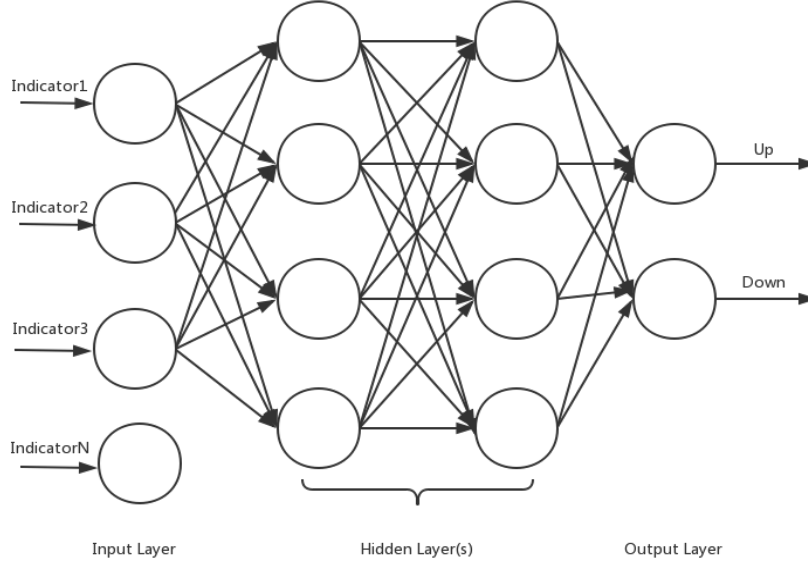


Figure 3: Example framework of multilayer perceptron

for hidden layer(s) between input layer and output layer, they accept vector from output by input layer or last hidden layer, and output vector to next hidden layer output layer, the process of which can be seen as finding correct mapping function. Each layer is fully connected to next layer, while within each layer there is no any connection. Except for neurons in input layer, every neuron in other layers has a non-linear *activation function*, which normally takes *sigmoid form*:

$$y(v_i) = \tanh(v_i) \text{ or } y(v_i) = (1 + e^{-v_i})^{-1},$$

where v_i denotes linear weighted inputs for every neuron.

MLP is especially outstanding in finding non-linear relationship between input and output, which makes it a quite useful tool in many jobs like classification and regression.

3.1.5 DAE (Denoising Autoencoders)

DAE is a variant of AE (Auto-encoder), a deep learning model. AE *encodes* input variables to a high-level representation and then tries to *decode* high-level representation to input variables, i.e. its original form. The high-level representation is believed to be a good simplification for inputs including too many redundant information, and is actually proved to be of better performance in replacing original input vector in later jobs [9]. As is shown in Figure 4, there can be many layers of hidden units, which means we can use AE to obtain feature representation that is abstract enough and vital enough to clear noises and redundant information contained in inputs. DAE adds *denoising* process to make AE more robust by randomly corrupting input vectors during training process, so that model is encouraged to capture the statistical dependencies between the inputs.

3.2 Models for Comparison

Inspired by Kim's classical work [5], here we also set our goal on assessment of precision of different models to predict the close price movement direction in near future.

Our models include ARIMA, LR, MLP, SVM and DAE-SVM. DAE-SVM is a assembled model of DAE and SVM which is inspired by Troiano's work [13]. Rather than have input features put into SVM's training process directly, input features would first be processed by DAE for dimension reduction and useless features' eliminating.

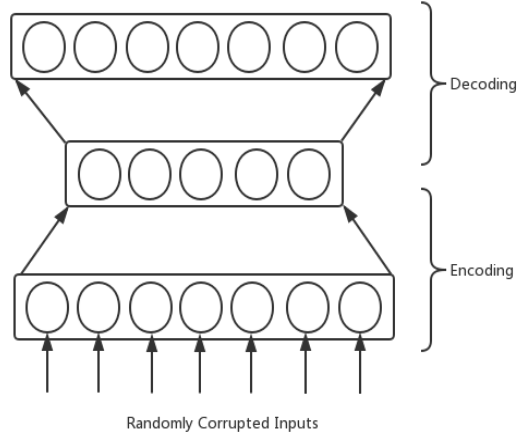


Figure 4: Simple Framework of Auto-encoder

While ARIMA only needs to extract information from past price time series, machine learning models and deep learning models need to take in certain features into model training, which can include features aside from price time series. Therefore, we decide to take technical indicators used in [5] as input features, which are listed in appendix A.

4 Experiment

4.1 Goal

This experiment aims to compare abilities between time series models and representative machine learning models to predict future trend of stock indices price. Based on this goal, the experiment referred to Kim’s work in introducing SVM into financial prediction [5]: in chosen data range, different models predict trends, and ratio of correct predictions would be compared between models.

4.2 Experiment Tools

We wrote python programs to implement the experiment, including models’ implementation, data sets’ preprocessing and models’ training and trend prediction. Our program runs on PyCharm 2017.

We use Python library *statsmodels* [10] API to implement ARIMA model, *sklearn* [8] to implement machine learning models, and *Theano* [12] to implement DAE.

4.3 Research Data

We choose S&P 500, Dow 30 and Nasdaq indices as our predicting data, which are representative international stock price indices and have been taken as experiment object in many similar research.

Our data sets are downloaded from Yahoo.Finance’ database. Data ranges from Jan. 2nd, 2012 to Dec. 26th, 2016. The choice of data range mainly depend on two factors. One factor is that models need enough amount of training data. The other factor is that we hope to use latest data, i.e. data from recent years.

Our data includes open price, close price, high price, low price and trading volume for the day. Taking short-time significant volatility into consideration, our unit time period is one week. For example, our close price are weekly close price, namely the close price of last trading day in a week, so we call it weekly close price. Three indices’ history weekly close price data is showed in figure 5.

After collecting data, we first preprocessed weekly close data, and obtain its trend series and take it as our goal series for testing models’ prediction precision. We deal with it in this way: For every weekly close price, if it’s greater than its value of last period, then trend for this week is set to 1, namely up trend. Conversely, set to -1, namely down trend. Apart from this, we write program to extract technical indicators from trading

data according to indicators' formulas, which would then be used for machine learning models' training and testing.

According to usual method to test machine learning model, we split training data into two continual parts. First part (70%) is for training the model, while the left part (30%) is for testing the precision of predicting, which is a common practice to split data in order to test the actual performance of machine learning model. For ARIMA, We split data the same way, while ARIMA only needs to model on weekly close price time series.

In order to ensure we get best machine learning models, i.e. models that are expected to have the best performance on test sets, we use cross validation when training models.

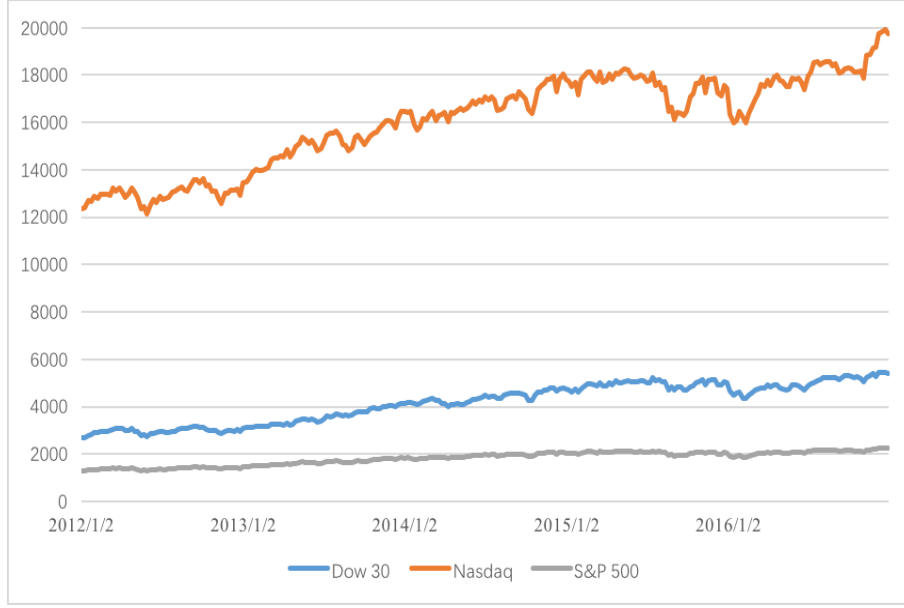


Figure 5: Weekly Close Price Series of Three Indices

4.4 Performance Measurement

Here, we use *hit ratio* to measure the precision of prediction, whose formulation is showed below:

$$hit\ ratio = \frac{\sum_{i=1}^N prediction_i}{N},$$

and

$$prediction_i = \begin{cases} 1 & prediction_i * real_i > 0 \\ 0 & else \end{cases}$$

where $prediction_i$ denotes the prediction of i th sample's price, and $real_i$ denotes the real value of that. When $prediction_i * real_i > 0$, it means prediction value and real value have same sign, which is regarded as a "hit".

4.5 Results

The hit ratio of different models are showed Figure 6 below:

The best performance of different models are showed in Table 1 below:

ARIMA	LR	MLP	SVM	DAE-SVM
0.593	0.623	0.63	0.642	0.672

Table 1: Best hit ratio of of different models

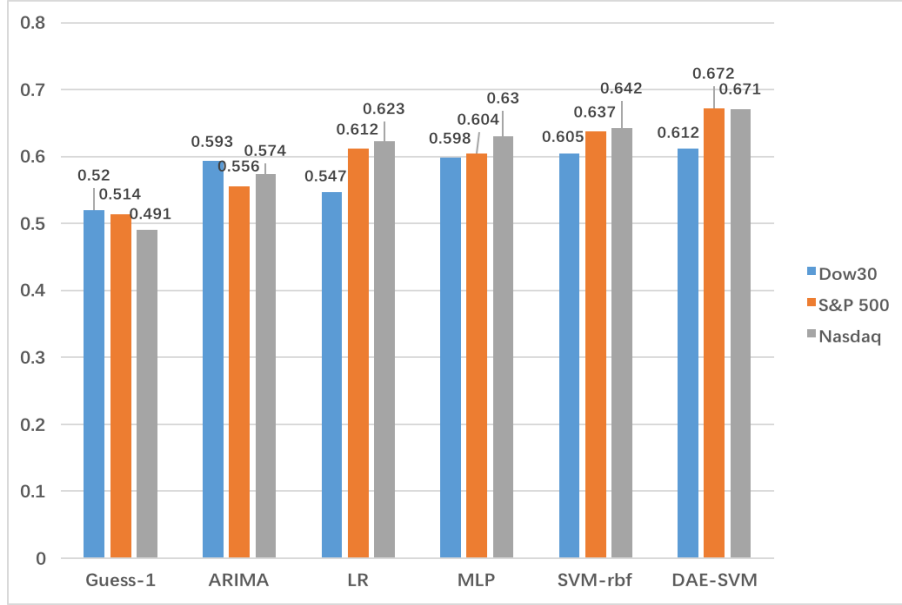


Figure 6: Hit Ratio of Different Models

Guess-1 represents a benchmark model that simply output 1 (predict future price to be up trend) for all test date, and its hit ratio is actually the percentage of up trends in test data. As what can be seen from Figure 6, expect for benchmark model Guess-1’s hit ratio keeping around 0.5, all other models attain hit ratio in range [0.5,0.7].

For differences between the time series model and machine learning models, all machine learning models perform obviously better than ARIMA model on different indices (Table 2), except for that LR’s hit ratio on Dow 30 indices is lower than ARIMA’s.

$\Delta(\text{Hit Ratio})$	LR-ARIMA	MLP-ARIMA	SVM-ARIMA	DAESVM-ARIMA
Dow 30	-4.5%	0.5%	1.2%	2%
S&P 500	5.6%	4.8%	8.2%	11.7%
Nasdaq	4.9%	5.6%	6.7%	9.7%

Table 2: Differences Between Performances of ARIMA and Machine Learning Models

Besides, while DAE-SVM and SVM performs better than all other machine learning models on different indices, DAE-SVM also surpasses single SVM completely, with highest hit ratio of 0.672 on S&P 500.

5 Conclusion

Our experiment results shows that while ARIMA as a representative model of traditional time series models in security price prediction is effective (better than the benchmark model), machine learning models attains generally better precisions in predicting. Besides, among all machine learning models in the experiment, SVM outperforms all other machine learning models, and can perform better with the combination of the deep learning model auto-encoder, because auto-encoder is a unsupervised learning model proved to be effective in extracting ‘higher-level’ representation of input features, so that it can assist other supervised learning models in enhancing their performances.

This paper is also limited in following respects: first, we choose input variables (technical indices) used for machine learning models training based on certain existing similar research, but there are also other researches where the researchers choose other input variables also attain similar or even better results on their data sets. Therefore, the criterion for choosing model input features still remains to be completed. Secondly, while there are several similar researches that choose different data sets, we choose international stock indices that are most frequently used in existing researches (Dow 30, Nasdaq and S&P 500), but there are still experiment factors like time range and sampling frequency that need to be considered when choosing

data sets, so it's still difficult to compare our models' performances with other researches', and this is a problem that we will try to solve in later research.

Besides, many researches give variants of SVM model and proved that the variant outperforms original SVM in many aspects, and therefore we can also try incorporating these variants into prediction experiment. And as is mentioned before, there are several deep learning models such as LSTM that have been proved to be quite applicable to deal with data with time series properties. Therefore, they can also be used in the experiment and may attain better performances.

References

- [1] M. Nasser M. Mufakhkharul Islam Altaf Hossain, Faisal Zaman. Comparison of garch, neural network and support vector machine in financial time comparison of garch, neural network and support vector machine in financial time series prediction. *Pattern Recognition and Machine Intelligence*, 2009.
- [2] Corinna Cortes and Vladimir Vapnik. *Support-Vector Networks*. Kluwer Academic Publishers, 1995.
- [3] J. B Heaton, N. G Polson, and J. H Witte. Deep learning in finance. 2016.
- [4] Dr. A. K. Mittra J. G. Agrawal, Dr. V. S. Chourasia. State-of-the-art in stock prediction techniques. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 2, 2013.
- [5] Kyoung jae Kim. Financial time series forecasting using support financial time series forecasting using support vector machines. *Neuro Computing*, 2003.
- [6] Julie R. Kirkpatrick, Charles D.; Dahlquist. *Technical Analysis: The Complete Resource for Financial Market Technicians*. Financial Times Press, 2006.
- [7] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [9] Bernhard Schölkopf, John Platt, and Thomas Hofmann. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems*, pages 1137 – 1144, 2006.
- [10] Skipper Seabold and Josef Perktold. Statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010.
- [11] T.N. Goh S.L. Ho, M.Xie. A comparative study pf neural network and box-jenkins arima modeling in time series prediction. *Computers and Industrial Engineering*, 2002.
- [12] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.
- [13] Luigi Troiano, Elena Mejuto, and Pravesh Kriplani. On feature reduction using deep learning for trend prediction in finance. *Papers*, 2017.
- [14] W. Shahzad Z. Mahmood Zahid Iqbal, R. Ilyas and J. Anjum. Efficient machine learning techniques for stock market prediction. *International Journal of Engineering Research and Applications*, 2013.