Price Prediction Evolution: from Economic Model to Machine Learning

By Hugh Ren (hughrxy), Xiangguang Zheng (zhengxg3), Erico Cruz Lemus (ecruzlem)

Abstract

This report is intended to illustrate timeline progress in the research on the trade market predictions using macroeconomic features to other different machine learning approaches. We design different algorithms, we prove them and we validate each result using as a reference the index S&P 500 as the main data source.

I. INTRODUCTION:

Latest decades have seen a rapid growth of the trade market. Part of this rapid growth has been driven because of the result of two significant factors, the reduction of trade barriers, zooming the rise in international trade, and the exponential improvement of technology such as the Internet, smartphones, etc.

The prediction of the stock market is a topic of interest, in particular for those who invest in it. It would be very useful to be able to predict the trend and, if possible, the price of the stocks, so with such information the investors could take relevant decisions that help them to obtain significant profits.

Consequently, for the last decades, price forecast has been tried to be predicted by multitude experts using different approaches.

Initially, given that the trade market and its closing prices can be treated as times series, most of the experts started to use basic statistic tools. In the decade of 90s, some experts tried to use macroeconomic data as some predicting features. Nowadays, experts are using more sophisticated statistical tools, including different machine learning approaches.

Unfortunately, most of the operations are originated by news, emotions, and other factors, making their behavior very unpredictable. We decided, then, to focus our work on the Low-frequency trading LFT, trying to predict closing price using different approaches, from using macroeconomic features, as some experts tried to do before, to using ML algorithms.

II. DATA ACQUISITION AND FEATURE SELECTION:

S&P500 price and index datasets are applied to run our algorithms. For LWR intuition, we only take opening stock prices of one company as datasets. For SVR and NN, we used shifting window pattern to exact the input data from stock index, which reflects a comprehensive market tendency, as shown in figure 1. To introduce more dimensions for the input data, we further aggregate the input into four features (max/min/ mean/SD) among every five days.

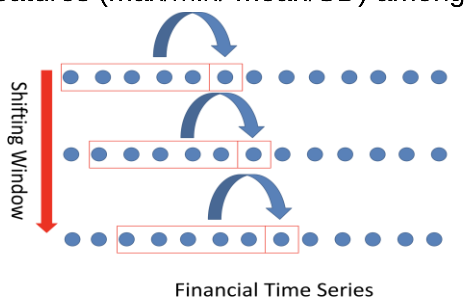


Figure 1

We run multilinear regressions adding each macroeconomic feature one by one to see the outputs predictions. The purpose of this method is to select different macroeconomic features.
We built a traditional multilinear regression that predicts today's close price based on the average of the previous 5 days' close prices, adding some major macroeconomic indicators, which included GDP (Gross Domestic Product), VIX (Volatility Index), and CPI (Consumer Price Index).
We chose GDP and CPI rather as exploration because these are the conventional macroeconomic variables that some expert used in their research in the decade of the 90s, and seem to plausibly relate to overall economic trends in close price variation.

III. MODELS

A. LWR:

Our first approach was using the Locally Weighted Linear Regression model (LWR). We want to first utilize this model to show some intuitions about how think and act on the stock market. Let us consider novices (they lack of effective information source and predicting knowledge) in stock market: on one day, if we give them money and require them of getting profit in stock market, how they will do? A simple idea is that they just review stock price and tendency in recent days, and approximately estimate a future price. That is what LWR do-it give exponential decreasing weight to recent day's loss function and make a prediction by minimizing the loss. In LWR, we omit other informative data in stock market and only consider stock price as the feature.
Before talk about results, it is necessary to discuss mathematical details (especially for weight) of LWR. First, let us talk about the weight matrix difference between predicting mediate price and future price. LWR is a non-parametric regression, so the output theta depends on relative coordinate of feature in the feature space. In a one-dimensional "time" space (since we focus on price-time relationship now), the weight only depends on the distance between t and t'. Based on this understanding, we are able to realize that: when predicting the mediate price, the weight matrix value in diagonal (actually only the entries in diagonal is not zero) is concave, while the weight matrix value in future prediction is monotonically decrease. It should be note that the property is helpful to understand the following results. Second, the concept of "shifting window" is introduced to update price data. Assuming that the training datasets are a time series (i.e. Ta to Tb) of stock price and we want to use the feature to predict the stock price of the following time series, what we need to do next? In our model, the dataset is updated in real time: we assume the investors will check their purchased stock every several days (3 days in our algorithm). The observation result will refresh the input set. According to our information, we believe that this assumption is quite realistic, providing, also, the right execution of the model because of the updated input information.

B. SVR:

Support vector machine (SVM), originally proposed by Vapnik in the 1990s, is a significant machine learning method in data mining, and combining with kernel trick, it has been an effective way to overcome difficulties such as curse of dimensionality and bias/variance. However, SVM is mainly used to solve the classification problem, while stock data distribute in a continuous space and predicting of stock behavior means forecasting the curve tendency (regression problems). Intuitively, it might be easy to convert a regression to a classification by cutting data space into several subspaces and mark it with label (just like how we discrete feature space of inverted pendulum in class). But that unavoidably introduce the ambiguity of predicting, and if we want to clarify it, we have to set substantial labels which might fail SVM. And modifying the loss and restrict function might be another alternative way. After reviewing reference, we find support vector regression (SVR) could do this by adding relaxation variables so as to make SVM "flexible". The initial loss function and restriction could be:

$$\min_{w,b,\lambda_1,\lambda_2}\left[\frac{1}{2}w^T w + C\sum_{i=1}^{n}(\lambda_1^i+\lambda_2^i)\right]$$

$$\begin{aligned} s.t.\quad & y_i - w^T\varphi(x_i) - b \le \varepsilon + \lambda_1^i \\ & w^T\varphi(x_i) + b - y_i \le \varepsilon + \lambda_2^i \\ & for\ all\ \lambda_1^i, \lambda_2^i \ge 0 \end{aligned}$$

Where ε is constant and all λ is relaxation variable, it appears in both target function (the larger λ is , the harder to minimize the target function) and restriction function (loose restrict boundary). So, the goal of solving the above is to construct a hyperplane that is as close to as many of the data points as possible. After that, we can solve this optimization problem in dual space with Lagrange multiplayer. Since we only apply it to regress the stock price, we will not discuss so much details (Debasish's paper about SVR is our reference and it contains detail discussion). By the way, Gaussian kernel is utilized in SVR thanks to its infinite dimension pattern.

C. Artificial Neural Network:

The artificial neural network we used has 2 hidden layers and ReLU as the activation function for hidden layers. And we didn't use any activation for the output layer in order to use ANN as regression.

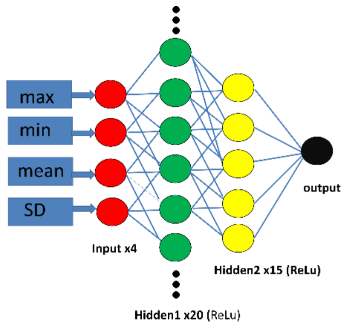


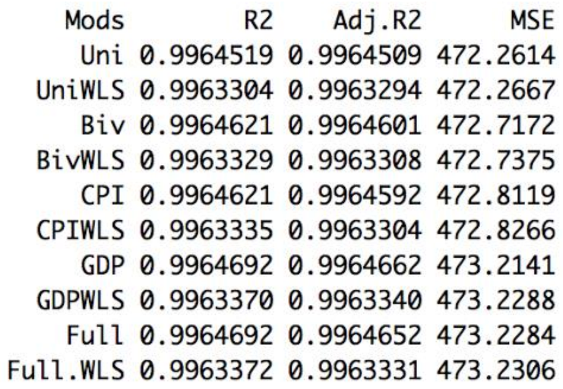
Figure 2

We also considered different activation function, where it turns out ReLU perform the best. ReLU provides multiple advantages compared to other activation function. First of all, it speeds up the training, the gradient computation is easy, and the computation step is quick, which is crucial when the input size is huge and when the topology of the network is complex, especially given the complexity of the training of the neutral network. Second, activation like sigmoid suffer from the vanishing gradient problem, where the gradient essentially becomes 0 after a certain amount of training and stops all training in that section of the network, this is due to the curve of the sigmoid function, where the output would be similar if the input of the sigmoid function is too big or too small.

IV. RESULTS

A. Macroeconomic Multilinear regression-features selection:

We built 10 separate multi regression models (some with weighted least squares approach) with varying combinations of the aforementioned variables.
In total we used 4 different input feature (univar, bivar, CPI and GDP). GDP are significant predictors of tomorrow's close, even after accounting for the previous 5 day's close as a variable. CPI does not significantly predict tomorrow's close.
As shown in the figure 3, adding more economic features worsen our predictions. This could be originated because of the nature of this data. Macroeconomic features values oscillate slower, in a long term, however trade market indexes closing prices are updated every day.



```
     Mods        R2    Adj.R2      MSE
      Uni 0.9964519 0.9964509 472.2614
   UniWLS 0.9963304 0.9963294 472.2667
      Biv 0.9964621 0.9964601 472.7172
   BivWLS 0.9963329 0.9963308 472.7375
      CPI 0.9964621 0.9964592 472.8119
   CPIWLS 0.9963335 0.9963304 472.8266
      GDP 0.9964692 0.9964662 473.2141
   GDPWLS 0.9963370 0.9963340 473.2288
     Full 0.9964692 0.9964652 473.2284
 Full.WLS 0.9963372 0.9963331 473.2306
```

Figure 3

We also believe that averaging the data of the five previous days in order to predict the fifth one, smooths the data so we could miss the real behavior of the trade market and its data.
Therefore, we believe that using closing prices with macroeconomic features, is not a good approach for obtaining accurate predictions in the trade market.

B. LWR

We built up the regression via MATLAB. Serving as a typical example, we use datasets from ASTRA to show our results. Before running, the parameter ‘t’ in exponential weight should be selected to avoid over/under fitting. As shown in the table, the t=5 might be a good choice.

Parameter t	1	5	10	100
Average-Error	1.3730	1.2065	1.3234	2.7314

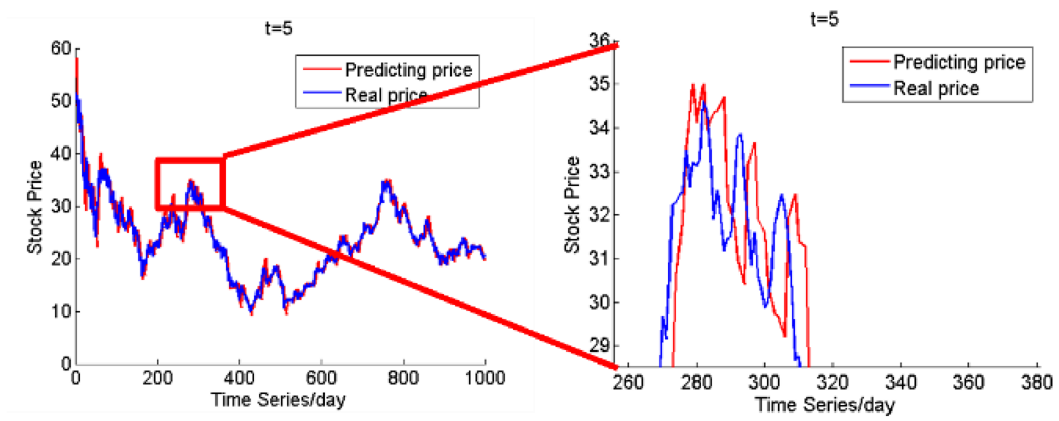


Figure 4

LWR present a good intuition of price prediction, but still with shortcomings. An obvious one is time lag: as shown in the above figure, if amplifying the whole predicting curve, we could find the predicting and real price have time lag. That is reasonable because of both lack of data preprocessing and LWR itself-if input data is a time sequence, LWR weight matrix is a constant, so LWR has poorly learned something, but just linearly combine previous price. Therefore, we still need another advanced model.

C. SVR

We use libsvm in MATLAB as our SVR library and divide 5000 preprocessed data into three groups: 4000(train)+500(dev) +500(test). The dev set is mainly used to optimize two parameters (C in target function is 10^{-11} and in Gaussian kernel is 10^{6}, that understandable because our data is not normalized). The output 500 test data with 384 MSE is compared with actual data and NN result in the following figure, and comparison of those two method is the discussion part.

D. Artificial Neural Network:

We use tensorflow as backend for running the ANN. For the 5000 preprocessed data input, we divide up the 5000 preprocessed data into three groups as we did for SVR: 4000(train)+500(dev) +500(test), where the dev set is used to adjust the model's topology, including the number of layers, number of neuron per hidden layer, as well as the size of mini batch. The result is shown in Figure 5, with MSE of 378.

V. COMPARISON

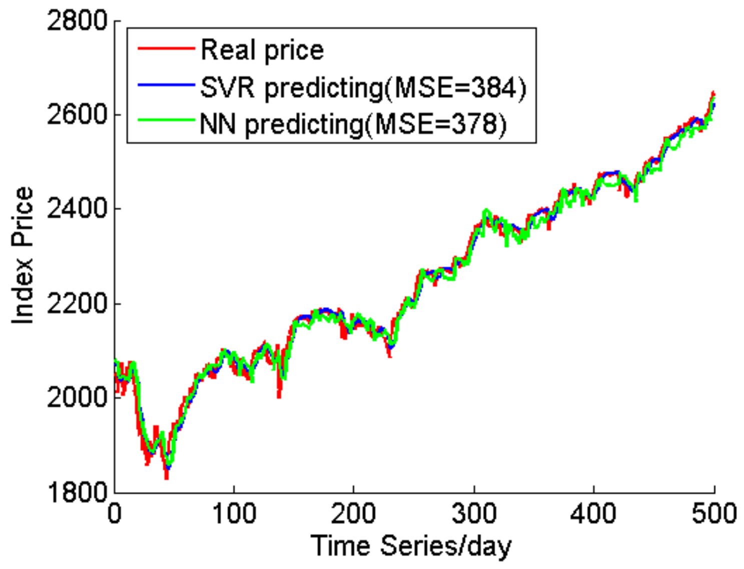


Figure 5

The figure 7 shows the prediction comparison between the SVR and NN model. To compare the SVR and neutral network. One to the cons of the NN is that NNs are heuristic, while SVMs are theoretically founded. Specifically, stochastic gradient descent isn't guaranteed to find the optimal set of parameters when used the way NN implementations employ it. However, any decent SVM implementation is going to find the optimal set of parameters. People like to say that neural networks get stuck in a local minima while SVMs don't. Like in our case, during the experiment, the result of NN is very unstable, ranging up to a few thousands MSE, causing by converging to a local minimum point. While the SVR is much more stable.

In addition, sensitivity shown in figure, NN always overreact in tiny fluctuation, which implies NN “prefer” to capture signal and endow high weight on it, while SVR behaves more smoothly on prediction.

But to some extent, NN has the ability to outperform the SVR when the model's complexity is high, which is, when the input dimension is high, enforce the model to increase complexity in order to fully capture the features from input. One specific benefit of ANN over SVR is that its size is fixed: it is parametric model while SVR is non-parametric model. It means that the number of parameters grows with the size of the training set for SVR, which causes much more burdon when the input feature has high dimension. While in the field of price prediction, we need to keep increasing the dimension of input space in order to eliminate the noises and therefore improve accuracy, which makes ANN a lower cost option.

VI. CONCLUSION AND FUTURE IMPROVEMENTS

SVR and NN are the final evolution version, and they output similar MSE comparing with actual data, while each hold various advantages and drawbacks. Our result indicates that the ANN is less stable than the SVR but provides more potentials.

If we had more time working on this interesting topic, we wil explore the followings:

1. Combine them with respect to optimized weight parameters to achieve a hybrid model with better accuracy.
2. Use RNN to better capture the input with the sense of time sequence and compare the performance with our models
3. Further extend our model to predict the optimal investment portfolio based on predicted price, which can be used in real world for financial trading.

VIII. CONTRIBUTIONS

Hugh Ren: LWR model, SVR model, poster preparation and presentation, final report.

Xiangguang Zheng: Artificial Neural Network model, final report.

Erico Cruz Lemus: Data research, macroeconomic regression feature selection, final report.

VI. REFERENCES

[1] M. E. Gerlow, S. H. Irwin and T. Liu. “Economic evaluation of commodity price forecasting models.” International Journal of Forecasting , vol. 9, pp. 387-397, 1993.

[2] J. Z. Wang, J. J. Wang, Z. G. Zhang, S. P. Guo. “Forecasting stock indices with back propagation neural network.” Expert Systems with Applications , vol. 38, pp. 14346-14355, 2011.

[3] C. O. Tiong, C.L. Ngo, and Y. Lee. “Stock Price Prediction Model using Candlestick Pattern Feature.” International Journal Of Interactive Digital Media, vol. 1(3), pp. 58-64, 2013.

[4] B. A. hnaity and M. Abbod. “Predicting Financial Time Series Data Using Hybrid Model.” Intelligent Systems and Applications, vol. 650, pp. 19-41, 2016.

[5] D. Basak1, S. Pal and D. C. Patranabis. “Support Vector Regression.” Neural Information Processing – Letters and Reviews, vol. 11, pp. 204-224, 2007