Using AI to Make Predictions on Stock Market

Alice Zheng
Stanford University
Stanford, CA 94305
alicezhy@stanford.edu

Jack Jin
Stanford University
Stanford, CA 94305
jackjin@stanford.edu

1 Introduction

In the world of finance, stock trading is one of the most important activities. Professional traders have developed a variety of analysis methods such as fundamental analysis, technical analysis, quantitative analysis, and so on. Such analytically methods make use of different sources ranging from news to price data, but they all aim at predicting the company's future stock prices so they can make educated decisions on their trading.

In recent years, the increasing prominence of machine learning in various industries have enlightened many traders to apply machine learning techniques to the field, and some of them have produced quite promising results. In this paper, we will focus on short-term price prediction on general stock using time series data of stock price.

2 Background & Related work

There have been numerous attempt to predict stock price with Machine Learning. The focus of each research project varies a lot in three ways. (1) The targeting price change can be near-term (less than a minute), short-term (tomorrow to a few days later), and long-term (months later). (2) The set of stocks can be limited to less than 10 particular stock, to stocks in a particular industry, to generally all stocks. (3) The predictors used can range from global news and economy trend, to particular characteristics of the company, to purely time series data of stock price.

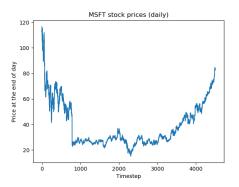
Based on the works we find, more progress has been made in predicting near-term [1] and long-term price changes [2]. In particular, long-term prediction has achieved over 70 percent accuracy when only considering limited number of stocks or sticks in a particular industry [3]. These well working models often rely on particular information of the company [4], which is often hard to access by general public and doesn't work with short-term prediction.

We decided to focus our project on the domain that currently has the worst prediction accuracy: short-term price prediction on general stock using purely time series data of stock price. Most researches in this domain have only found models with around 50 to 60 percent accuracy. And they often work only for classification [5]. After close investigation, we noticed that some papers that claim to have better result used problematic metrics to evaluate the model [6][7]. Although the graphs or numbers might look fancy, the actual model provides limited meaningful guidance for trading. (One example would be trying to predict whether the price tomorrow is increased or decreased from several days ago.)

Our preliminary model suggests that naively passing in time series data of stock price works no better than random guessing, so we applied complicated models to pre-process the time series data before running ML models. We didn't find any works that combined these financial technical indicators with machine learning algorithms like we did. And we got better results than other papers we found in this particular problem domain.

3 Dataset and Features

We use Alpha Vantage API [8] to access the time series data of 82 randomly chosen stocks traded at NYSE. The API provides access to intraday time series, daily time series, weekly time series and monthly time series data. Since the domain of our problem is short-term prediction, we decided to proceed with daily time series data, which includes daily open price, daily high price, daily low price, daily close price and the daily volume.



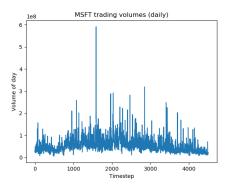


Figure 1: Daily prices and trading volume of Stock "MSFT"

Then we consulted several investment education websites. We carefully selected 11 models of processing time series data and get 17 technical indicators out of the models. The models are selected based on their popularity and reviews on the investment education websites.

APO-SMA Absolute price oscillator values with SMA	PO = SlowMA(price) - FastMA(price)	Moving Average Convergence Divergence (MACD)	$shortema = 0.15 \times price + 0.85 * shortema_{[-1]}$ $longema = 0.075 \times price + 0.925 * longema_{[-1]}$ $MACD = shortema - longema$ $(lose - LowestLow_{[loss n prish]} - LowestLow_{[loss n prish]} - LowestLow_{[loss n prish]}$		
APO-EMA		Stochastic oscillator (STOCH)			
CCI	$CCI = \frac{TP - ATP}{0.015 \times MD}$ $TP = \frac{high_n + low_n + close}{3}$ $TP = Typical Price high_n = Highest high in the last n time periods low_n = Lowest low in the last n time periods ATV = Simple Moving Average(TV) MDTV = Mean Deviation(TV)$		%D = MovingAverage(%K)		
Commodity channel index values		Relative strength	If $close > close_{[-1]}$ then		
		index (RSI)	$up = close - close_{[-1]}$		
		values	dn = 0		
			else up = 0 $dn = close_{[-1]} - close$ $upavg = \frac{upavg \times (n-1) + up}{n}$		
AROON	$AroonUp = 100 \times \left(\frac{n - PeriodsSinceHighestHigh}{n}\right)$				
	$AroonDown = 100 \times \left(\frac{n - PeriodsSinceLowestLow}{n}\right)$				
Bollinger bands (BBANDS) values	$TP = \frac{high + low + close}{3}$		$dnavg = \frac{dnavg \times (n-1) + dn}{n}$		
	MidBand = SimpleMovingAverage(TP) $UpperBand = MidBand + F \times \sigma(TP)$		$RMI = 100 \times \frac{upavg}{upavg + dnavg}$		
	$LowerBand = MidBand - F \times \sigma(TP)$	On balance volume	If $close > close_{[-1]}$ then		
Chaikin A/D line (AD) values.	$CLV = \left(\frac{(close - low) - (high - close)}{(high - low)}\right)$ $AD = AD_{t-1} + CLV \times volume$	(OBV) values.	$OBV = OBV_{[-1]} + volume$		
			elseIf $close < close_{[-1]}$ then		
Average directional	$ADX = \frac{ADX_{[-1]} \times (n-1) + DX}{n-1}$		$OBV = OBV_{[-1]} - volume$		
movement index			e1se		
(ADX) values.	n		$OBV = OBV_{[-1]}$		

Now we put together the stock price at the end of the day at the beginning of each period, the volume of the same day, and the technical indicators above and we have 260k data samples from 82 stocks. Together with the daily trading volume and the price, we have 19 predictors in total.

4 Methods

The problem we want to tackle is hard, so we started with a simplified problem: predicting whether the prices will increase or decrease in the next n days, using the stock prices and volumes in the past mdays. For this classification problem, we implemented Logistic Regression, Bayesian Network, Simple Neural Network, and SVM with rbf kernel in the sklearn [9] library and ran them on the prices of one specific stock named "MSFT".

After getting our preliminary results, we included the technical indicators in the predictor and tried to predict the exact change in prices in the next n days. We reserve the stock "AEB" for presenting our results, and use the data of other stock for training and testing. For every trial, we randomly shuffle the data and partition them into training set, and test set in a ratio of 9:1. Then we use two metrics to evaluating different regression models: the mean squared error, and the prediction

error rate (whether the price will increase or decrease). The first metric is primarily used for checking whether we have an issue of over-fitting since it's not as straight-forward and cannot be generalized to different data with different variances. The second one is our primary criteria and is used for comparing our results to our preliminary findings.

We started with Linear Regression and applied two different regularization of linear regression, Lasso and Ridge, to see whether we can improve our results. However, we don't expect over-fitting to be a big problem, so we expect such regularization will not give us much improvement over the basic model, but they can still be useful in checking whether our predictors are significant.

Then we applied Support Vector Regression (SVR) with radial basis kernel. Although this model performs really well, we cannot run it on entire data set due to our limited computing power. The space complexity is $O(n_observations^2)$ in order to apply the kernel trick, which is readily problematic. Furthermore, the time complexity of the model is also very high. Therefore, we ran the model only on a subset of our data.

5 Experiments and Discussion

5.1 Preliminary Experiments

Before we approach the actual problem we want to tackle, we start with a simplified question: can we actually predict the prices trends? We implemented several models available in the sklearn package and the following are our results on the stock 'MFTS':

(m, n)	(20, 5)	(20, 3)	(10, 3)	(10, 1)	(5, 1)
Logistic Regression	51.26%	51.04%	52.37%	47.97%	48.08%
Bayesian Network	50.84%	50.97%	48.52%	47.09%	46.87%
Simple Neural Network	47.06%	45.93%	44.66%	42.14%	42.83%
SVM with rbf kernel	46.22%	44.79%	43.38%	41.33%	41.01%

Table 1: Test error rates for different models

From the results shown above, we can see that the models we implemented are close to guessing randomly since most of the error rates are around $40\%\sim50\%$. This suggests that the underlying process is not well represented by the models we chose, or our choice of predictors are not good. Neural networks with sigmoid activation only have a mediocre performance, suggesting that there are not a very high amount of non-linearity in the actual underlying model. Also, we see that SVM with a radial basis kernel gave the best performance, suggesting that there might be underlying classes for our sample and each class could have its own model.

Comparing the results across different (m, n) pairs, we also see that using a larger m does not necessarily lead to a smaller test error rate on the test set, so only the most recent prices are more meaningful in the prediction. Also, it is easier to predict the price trend for the next day than 3 days later, suggesting that forecasting prices in the long future might not be possible.

5.2 Linear Regression with Technical Indicators

Our preliminary results were sufficient to show that past prices by itself is not a good predictor for the problem we would like to tackle, so we introduced more predictors into the problem. After doing research [10][11] we found out that in the field of trading, people often use technical indicators for making decisions, so we decide to include them in our model. We chose the 13 most used indicators in the field and together with the stock prices we have a total of 19 predictors for time-step. Using these 19 predictors, we attempt to predict the price change for the next n days.

From the results of our preliminary findings, we decided to start with linear regression.

For the first metric we discussed in the methods section, we see that the test loss oscillates between 0.2 and 0.8 for n=1 with different random partitions of the data, while the training loss stays around 0.5 to 0.6. This suggests that over fitting is not an issue for our model and the model generalizes quite well.

For the second metric, we achieved a correct prediction rate as high as 61.5% when n=1, and around 55% for $n \ge 10$. Figure 2 is a diagram of the correct classification rate.

We see that the results are consistent with the results of our preliminary findings, and compared to those results we get more improvements on the prediction for when n is big, but not as much improvement for smaller values of n.

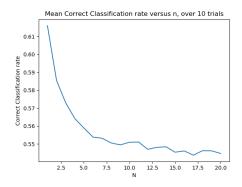


Figure 2: Correct Classification rate for linear regression

Besides plain linear regression, we also implemented ridge regression and lasso regression to justify our selection of the predictors. An output of the coefficients of ridge regression reveals that no coefficient has dramatically decreased. This suggests that all our predictors are significant, which is again justified by comparing the test error rates listed in Figure 3.

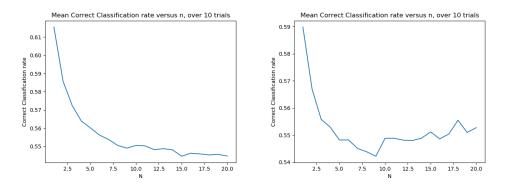


Figure 3: Ridge Regression and Lasso Regression

5.3 Support Vector Regression

From our preliminary findings, we know that SVM with a Gaussian kernel performs well, so we decided to implement support vector regression on the data to see whether we can get a better result than those of linear regression. However, in the implementation we were forced to work with a smaller training set due to the nature of the model: Since Gaussian kernel maps our predictors into an infinite dimensional feature vector, we are forced to apply the kernel trick, which leads to a space and time complexity of $O(n_{samples}^2)$. Working with 200k+ samples is therefore not an option. In our experiment, we randomly select 5% of our training data as the data we use for the regression and we produced the following error rates shown in Figure 4.

Unlike linear regression, we see that the error rates oscillates since we choose a smaller training set, leading to more variance in our fitted model. a common trait, however, is that we see the same pattern as n increases, since the correct rate drops around n=20. Generally, we see that we can achieve a correct rate of 69.5% for the highests and 68.5% for the lowests, which is a big improvement on linear regression, and also superior to many similar projects. [5][6][7]

5.4 Other Attempt

Besides the models we discussed above, we also implemented a neural network with a hidden layer based on LSTM (long-term-short-term memory) and a dense layer for output. The model is widely used for various NLP tasks such as voice processing, and its incorporation of the concept of keeping memory of past data in a time series suggest that it could be useful for stock price prediction. [12] Unfortunately, however, our implementation of the model was not promising since we saw that the development MSE converges around the variance of the development set, with the converged model outputting predictions close to 0 and a correct prediction rate of 50%.

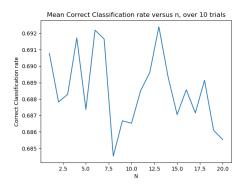


Figure 4: Support Vector Regression

6 Conclusion and Future Work

Concluding from our results, we see that using only past price data and technical indicators which are calculated based on past price data, we can achieve a correct prediction of the price trends for a shy of 70% of the time. Incorporating other factors such as news about the stock market and the companies will certainly improve the performance of our current model.

Although a correct rate of 70% is not very high, we can still see that the results can be quite meaningful. Below is a chart of the predicted price change and actual price change for the next day:

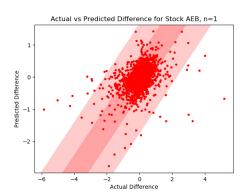


Figure 5: Prediction versus Actual Price Change over 1 day

In practice, since small fluctuation might not by that meaningful, we are more interested in capturing the price trend when the actual price of the stock changes a lot (say more than \$2). We see that we can in fact achieve a correct classification rate of 92.3% for when the price increases by more than \$2, and 72.9% for when the price decrease for more than \$2.

For all the models we use, as discussed in the experiments, the error rate for test set and error rate for training set has been really close. Thus overfitting shouldn't be a problem for us.

On the other hand, we are more likely to take action when the predicted change of price is big (again say more than \$2). When we predict that the price will decrease by more than \$2, 97.3% of the time the actual price will decrease. Unfortunately we rarely predict that the price will increase by more than \$2: we only predicted that the price will increase by more than \$2 for 0.1% of the test data, so the sample is too small to produce a meaningful judgment.

For future work, the first thing we would do is definitely find machines with higher computation power to run Support Vector Regression on the whole data set to improve the accuracy rate.

At the same time, although our attempt with neural networks so far wasn't successful, there are sources [13] suggesting that artificial neural networks are useful for stock price predictions, so one of the future directions will be to implement more models based on neural networks.

Appendix

Contributions

Both team members were involved in all stage of the projects. The background research and analysis of related work was shared evenly. The processing of Data Set and selection of Features (Technical Indicators) was led by Alice Zheng. The implementation of Machine Learning Models was led by Jack Jin. The analysis of result was done by both members together.

References

- [1] Rechenthin, Michael David. Machine-learning classification techniques for the analysis and prediction of high-frequency stock direction. The University of Iowa, 2014.
- [2] Milosevic, Nikola. "Equity forecast: Predicting long term stock price movement using machine learning." *arXiv* preprint arXiv:1603.00751 (2016).
- [3] Shen, Shunrong, Haomiao Jiang, and Tongda Zhang. "Stock market forecasting using machine learning algorithms." Department of Electrical Engineering, Stanford University, Stanford, CA (2012): 1-5.
- [4] Bonde, Ganesh, and Rasheed Khaled. "Extracting the best features for predicting stock prices using machine learning." Proceedings on the International Conference on Artificial Intelligence (ICAI). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2012.
- [5] Leung, Carson Kai-Sang, Richard Kyle MacKinnon, and Yang Wang. "A machine learning approach for stock price prediction." Proceedings of the 18th International Database Engineering & Applications Symposium. ACM, 2014.
- [6] Hegazy, Osman, Omar S. Soliman, and Mustafa Abdul Salam. "A Machine Learning Model for Stock Market Prediction." arXiv preprint arXiv:1402.7351 (2014).
- [7] Dai, Yuqing, and Yuning Zhang. "Machine Learning in Stock Price Trend Forecasting." (2013).
- [8] Vantage, Alpha., Alpha Vantage Free Apis For Realtime And Historical Financial Data, Technical Analysis, Charting, And More! Alphavantage.co. N.p., 2017. Web. 19 Nov. 2017.
- [9] Pedregosa et al., Scikit-learn: Machine Learning in Python. JMLR 12, pp. 2825-2830, 2011.
- [10] Technical Indicator Reference. Fmlabs.com. N.p., 2017. Web. 19 Nov. 2017.
- [11] Investopedia. Investopedia. N.p., 2017. Web. 19 Nov. 2017.
- [12] Chen, Kai, Yi Zhou, and Fangyan Dai. "A LSTM-based method for stock returns prediction: A case study of China stock market." *Big Data (Big Data)*, 2015 IEEE International Conference on. IEEE, 2015.
- [13] Kyoung-jae Kim, Ingoo Han. "Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index". Expert Systems with Applications, Volume 19, Issue 2, 2000, Pages 125-132, ISSN 0957-4174.