ICS635 Machine Learning
Final Project:
Stock Price Prediction Systems

Song Min Kim, *ICS, UHM,* Hualiang Li, *EE, UHM,* and Akira Nishibayashi, *MBA, UHM*

***Abstract*—In this report, we used three machine-learning algorithms with daily data we gathered in order to conduct an experiment to see if our algorithms could accurately predict future S&P 500 price change. For this experiment, we referenced the Tensorflow [10] and Scikit-learn libraries [7] [8] [9].**

***Index Terms*—Machine Learning, Suport Vector Machine, Decision Tree, Feed Forward Artificial Neural Network, Stock Price.**

I. Introduction

THE purpose of our final project was to get a wide range of practical knowledge of and a deep understanding of machine learning. For decades, there have been many attempts to predict stock market's price by companies and by individuals too. They used their own experience and the experience of others, public knowledge, and research from academic intuitions. In spite of their efforts, many were unsuccessful achieving their desire results.

The stock market price data is serialized data by date and time. So, we thought that it is possible to predict one of the stock market prices by machine learning technologies. For our experiment, we used market data from Quandl Financial and Economic Web Service. As previously mentioned, we decided to predict S&P500, the one of most famous stock indices in the world. And while we did not try to predict S&P 500 actual stock price but we did attempt to predict if S&P 500 would close up or down on a given day. We used these three machine learning algorithms: Support Vector Machine, Decision Tree, and Feed Forward Artificial Neural Network. We also referenced Google Cloud's financial data analysis video and document for our project [3].

II. Dataset

WE used various types of indices that were supported Quandl financial and economic web service because Quandl, a Canadian technology company backed by some of world's leading venture capital firms, supports many economic/financial datasets and APIs [6].

A. Type and period of the dataset

We used daily data for various kinds of indices that seem to be related to S&P 500, such as foreign exchanges, stock indices, and commodities. In terms of stock indices, the dataset includes not only closed prices but open, highest, and lowest prices also because we believed that such a wide array of data would represent a more detailed and complete picture of various markets' conditions. The number of date of datasets total 3817 and were gathered from January 5, 2000 to October 28, 2016.

Foreign Exchanges	USD/CHF, USD/EUR, USD/JPY, USD/GBP, USD/CAD, and USD/AUD
Commodities	Gold and Crude oil prices
Stock Indexes	NASDAQ Composite (US), S&P500 (US), Dow jones industrial average (US), All Ordinaries (Australia), DAX (Germany), Hang Seng (Hong Kong), and Nikkei225 (Japan)

TABLE I: Dataset

B. Adjustment of the dataset

We had to make necessary adjustments to our dataset due to the time differences. For example, Japanese stock markets (i.e. Nikkei225) close for the day 14 hours before S&P500 closes. Therefore, in terms of other countries stock indices (All Ordinaries, DAX, Hang Seng, and Nikkei225), we could use the same date's data as S&P500 in order to predict S&P 500. However, in terms of other American indices such as NASDAQ Composite, and Dow jones industrial average, they move at the same the time as S&P 500, so we could not use the same date's data for predicting S&P 500 and have to use the data one day ago.

Another issue we face were the different scales of the markets. The difference of scales created a different mean and standard deviation. Therefore, we used percentage change compared to the previous day for modeling our algorithms that have similar mean.

III. Algorithms

A. Methods of evaluation

We used 80% of our data as training data and 20% as test data. Our algorithms train the model by using the training data and check Accuracy and F1 score

$$F1\,score = 2 * Recall * Precision/(Recall + Precision) \quad (1)$$

for test data. Accuracy shows the percentage our algorithms correctly predicted in the test data.

F1 score is also a measure of tests accuracy. Best F1 score is 1 that means the system perfectly forecasts the results and

Dr. S. Still is with the Department of Information and Computer Science, University of Hawaii at Mānoa, Honolulu, HI, 96822 USA e-mail: (see http://www2.hawaii.edu/˜sstill/contact.html).

worst score is 0 that indicates the system does not work at all. In general, the recall of systems with high precision is low, and conversely, in systems with high precision, recall tends to be low. When there are two evaluation indices, it is difficult to compare which system is superior, so F1 score obtained by a weighted average of the recall and the precision was used as an indicator of the overall model performance.

At first, we tried to use the cross validation in order to evaluate out-of-sample error of our algorithms, but we thought that the cross validation method was not appropriate to our project and stock prices. It was because we felt it is problematic to check out-of-sample error of test data by the model that is trained by the future data. Therefore, we did not use the cross validation method to evaluate our algorithms.

B. Input data we used for the algorithms

We used data of the set (USD/EUR, DAX, Hang Seng, Dow jones industrial average, Nikkei 225) as the basic data set. We also used other data sets and analyzed the results. However, when there is no description about input data, input data is this basic data set in the following sections.

C. Support vector machine

We predicted the S&P 500's "up-or-down closings" with a support vector machine. The machine creates hyper-planes that have the largest distance to the nearest data points of any classes in dimensional space that is used for classification - the S&P 500 will close up or down.

In order to compute optimal separating hyperplane in feature space, we used kernel methods and tried to utilize some types of kernel such as linear, polynomial, sigmoidal, and radial-basis function in order to find the best-fit kernel to predict S&P 500. We also tried to adjust the cost parameter that controls between capacity of maximizing the margin and minimizing the margin error.

The following is the detail of support vector machine algorithm [9]. If training vectors are given as, $x \in R^p, i = 1, 2, \ldots, n$ and a vector $y \in \{1, -1\}^n$, the support vector machine solves the problem:

$$min_{\omega,b,\xi} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{n}\xi_i \tag{2}$$

subject to

$$y_i(\omega^T\Phi(x_i) + b) \geq 1 - \xi_i, (\xi \geq 0, i = 1, 2, \ldots, n) \tag{3}$$

ξ_i are slack variables that make the constraints relax and C is the cost parameter. The dual representation of the problem is

$$min_\alpha \frac{1}{2}\alpha^T Q\alpha - e^T\alpha \tag{4}$$

subject to

$$y^T\alpha = 0, (0 \leq \alpha_i \leq C, i = 1, 2, \ldots, n) \tag{5}$$

α_i are dual variables. Where e is the vector of all ones, $C > 0$ is the upper bound, Q is an n by n positive semidefinite matrix,

$$Q_{ij} \equiv y_i y_j K(x_i, x_j) \tag{6}$$

where

$$K(x_i, x_j) = \phi(x_i^T\phi(x_j)) \tag{7}$$

is the kernel. By using the kernel, training vectors are mapped into a higher dimensional space. The decision function is

$$f(x) = sgn(\sum_{i=x}^{n} y_i\alpha_i K(x_i, x) + b) \tag{8}$$

D. Decision Tree

The second model we used for our prediction was decision tree learning. The algorithm we used for this model was as follows [2].

Decision trees are produced by algorithms that identify various ways of splitting a data set into branch-like segments. These segments form an inverted decision tree that originates with a root node at the top of the tree. The object of analysis is reflected in this root node as a simple, one-dimensional display in the decision tree interface. The name of the field of data that is the object of analysis is usually displayed, along with the spread or distribution of the values that are contained in that field.

A sample decision tree is illustrated in Figure 1 [2], which shows that the decision tree can reflect both a continuous and categorical object of analysis. The display of this node reflects all the data set records, fields, and field values that are found in the object of analysis. The discovery of the decision rule to form the branches or segments underneath the root node is based on a method that extracts the relationship between the object of analysis (that serves as the target field in the data) and one or more fields that serve as input fields to create the branches or segments. The values in the input field is used to estimate the likely value in the target field. The target field is also called an outcome, response, or dependent field or variable.

Once the relationship is extracted, then one or more decision rules can be derived that describe the relationships between inputs and targets. Rules can be selected and be used to display the decision tree, which provides a means to visually examine and describe the tree-like network of relationships that characterize the input and target values. Decision rules can predict the values of new or unseen observations that contain values for the inputs, but might not contain values for the targets.

E. Feed Forward Artificial Neural Network

We also developed the feed forward artificial neural network algorithms (No hidden layer model, one hidden layer model, two hidden layer model) to predict if S&P 500 will close up or down with reference to "TensorFlow Machine Learning with Financial Data on Google Cloud Platform" [3]. In all models, all weights were initialized by using normal distribution and biases were initialized to one. The models were trained and

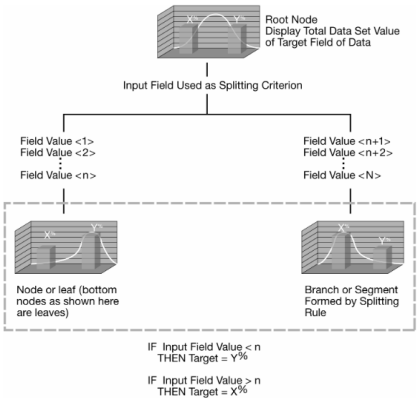

Fig. 1: Illustration of the Decision Tree [2]

the weights in our models were uploaded by back propagation with a stochastic gradient descent method called "Adam" [5] [1](Adaptive Moment Estimation). The number of iteration of the training was 10000.

Adam has many parameters: Exponentially decaying average of past gradients(m_t), exponentially decaying average of past squared gradients (v_t), time step (t) that is initialized to zero and increased one by one by each time ($t = t + 1$), constants that decides how fast the method forgets past information ($\beta_1 \, and \, \beta_2$), learning rate($\alpha$), and a parameter that prevents the denominator of the update amount from becoming zero when updating the weight(ϵ). At first, the method gets gradients

$$g_t = \frac{\partial C}{\partial w_t} \tag{9}$$

and then calculates m_t and v_t,

$$(m_t = \beta_1 m_{t-1} + g_t(1-\beta_1), v_t = \beta_2 v_{t-1} + g_t^2(1-\beta_2)) \tag{10}$$

They counteract these biases by computing bias-corrected first and second moment estimates

$$\hat{m} = \frac{m_t}{1-\beta_1}, \hat{v}_t = \frac{v_t}{1-\beta_2} \tag{11}$$

or update learning rate

$$\alpha_t = \frac{a\sqrt{1-\beta_2}}{1-\beta_1} \tag{12}$$

Finally, the method upload the weights

$$w_t = w_{t-1} - \frac{a\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{13}$$

[5], [1]. We used the sigmoid function

$$f_{(u)} = \frac{1}{1+e^{-u}}, (\frac{df(u)}{du} = f_{(u)}(1-f_{(u)})) \tag{14}$$

for the activation function in the output layer. The function expresses the probability of S& P 500 going up and down by utilizing outputs of the previous layer. In terms of hidden layers, the rectified linear unit function

$$ReLU : f_{(u)} = max(0,u), \left(\frac{df_{(u)}}{du} = \{1[0 \le u] \, or \, 0[0 > u]\}\right) \tag{15}$$

was used for the activation functions. In order to aim at a state where the outputs from the output layer becomes a more appropriate value against the input data of this network, we defined an error function by utilizing cross entropy.

$$C = -\Big(\sum_i y_i \log(o_i)\Big) \tag{16}$$

o is our algorithms' output, y is the real S&P 500 price change (if S&P 500 goes up, $y = [1,0]$, if S&P goes down, $y = [0,1]$). The cross entropy shows us how our model fits to the real S&P 500 price change and cross entropy will be close to 0 when all our outputs are close to the real S&P 500 price change. Cross entropy is always positive. We used cross entropy as an error function because the cross-entropy cost function can avoid the problem of learning slowdown in backpropagation, unlike the quadratic cost function.

IV. Analysis

A. SVM

1) Base Information: In this section, we used information as TABLE II,III,IV [8] [9].

Radial-basis function	$exp(-\gamma\|x-x'\|^2)$
Linear	$< x, x' >$
Polynomial	$(\gamma < x, x' > +b)^d$
Sigmoid	$tanh(\gamma < x, x' > +b)$

TABLE II: Kernel Function

Degree(d)	3
Independent term(b)	0
Gamma(γ)	0.06

TABLE III: Parameters of Kernel Functions

Cost parameter	range from 1 to 100
Input data	USD/EUR, DAX, Hang Seng, Dow jones industrial average, Nikkei 225

TABLE IV: Dataset

2) Kernel Function: Intuitively, a kernel is just a transformation of input data that allows us to treat and to process it more easily. Imagine that we have the toy problem of separating the 'x's from the 'o's on a plane as shown below Fig. 2 [4].

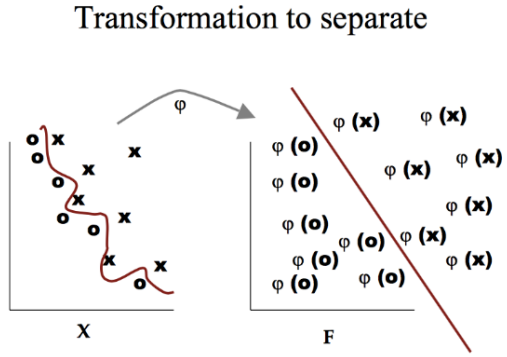

Fig. 2: An example of mapping to ease the classification

Our separating surface would be the curve drawn on the left figure. However, transforming our data into a different

space through the mapping (alpha function) shown in the figure would make the problem much easier since, now, our points are separated by a simple plane. Note that, it is possible to map 2-dimentions to N-dimensions if it is necessary. This embedding on a higher dimension is called the kernel trick. A kernel consists on embedding general points into an inner product space.

In our experiment, we tried four different kernel functions, which includes "Radial-basis function", "Linear", "Polynomial", and "Sigmoid". The result of accuracy was shown below in the table. From which, we conclude that for our stock price prediction model, the "linear" kernel function was among the best (with 68% accuracy and 0.72 F1 score). The fact that the linear kernel function had the highest performance was considered to be that the input data of indices could be linearly separated well in input domain. Using other kernel functions makes it possible to classify data in higher dimensional spaces that cannot be linearly separated in input domain, but in this case the linear kernel function seemed to be the most suitable for forecasting S&P 500. We thought the reason for this was that the number of features of our input data set was enough to linearly separate the data and mapping our data into higher dimensional spaces did not work to improve the performance. Therefore, we chose "linear" kernel function as our kernel function for IV-A3 changing cost parameter in the SVM.

TYPE	Radial-basis	Linear	Polynominal	Sigmoid
F1 SCORE	0.72	0.72	0.69	0.69
ACCURACY	66%	68%	53%	53%

TABLE V: Performance for different kernel function

3) Changing cost parameter in the SVM: The cost parameter controls the tradeoff between the capacity of maximizing the margin and minimizing the margin error. If the margin error is important, we should use a large value of the cost parameter. For a large value of the cost parameter, the machine will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, If the margin error is not so important, we should use a small value of the cost parameter. a very small value of the cost parameter will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points. For very tiny values of the cost parameter, one should get misclassified examples, often even if one's training data is linearly separable.

In our experiment, we tried different cost parameter values ranging from 1 to 100. The F1 score and accuracy results are shown in the chart below. The chart shows us that when the cost parameter was too low (below 20), our prediction result was suboptimal. It was because the small cost parameter has small power to minimize the margin error and tries to find separating hyperplane that has large margin, even if that hyperplane misclassifies more points. Moreover, from the graph, we see that once the cost parameter value reaches a certain value (approximate 20) in our model, the quality of our result almost stay same.

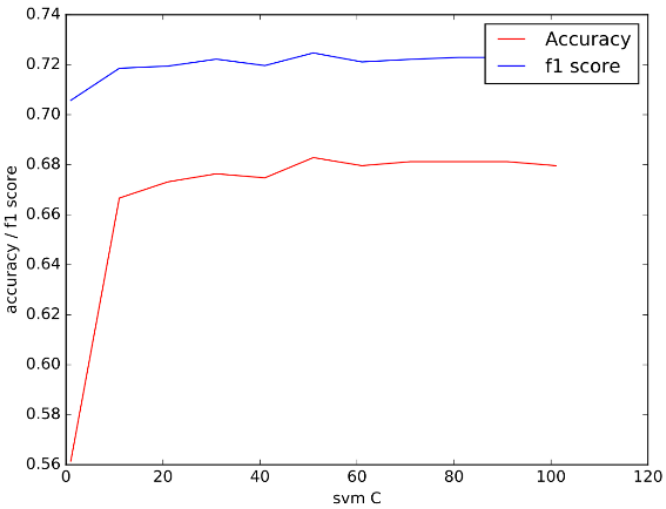

Fig. 3: Performance for different cost parameter

B. Decision Tree

1) Base Information: Iterate all different combination of parameters, and pick the best combination of parameters TABLE VI [7].

Maximum Depth	1 to 10
Minimum Samples Split	1 to 10
Minimum Samples Leaf	1 to 10
Input Data	USD/EUR, DAX, Hang Seng, Dow jones industrial average, Nikkei 225

TABLE VI: Decision Tree Base Information

2) Changing types of parameter: In this model, we customized three different parameters: maximum depth, minimum samples split, and minimum samples leaf. The maximum depth indicates how deep the tree could be in the worst case. The minimum splits correspond to the outcome of a test; it also connects to the next node or leaf. The minimum leaf terminal nodes that predict the outcome. In our experiment, by changing parameter of number of split and leaf, it did not affect our prediction performance. Therefore, we did not plot the performance graph. When changing the maximum of the decision tree, we found that when tree depth was lower than 4, we had a better performance than depth higher than 5 (See Fig.4. Performance for different tree depth). With tree depth of 4, our prediction performance achieved its best result. While as tree depth increasing from 5, the performance drops dramatically. We believed that this was because there were some critical indices and criteria to predict S&P 500 accurately. Therefore, the decision tree did not need high depth. If depth was higher than 4, the decision tree started to do overfitting and it harmed accuracy for our test data.

C. Feed Forward Artificial Neural Network

1) Base Model: In this section, we defined base model case as TABLE VII

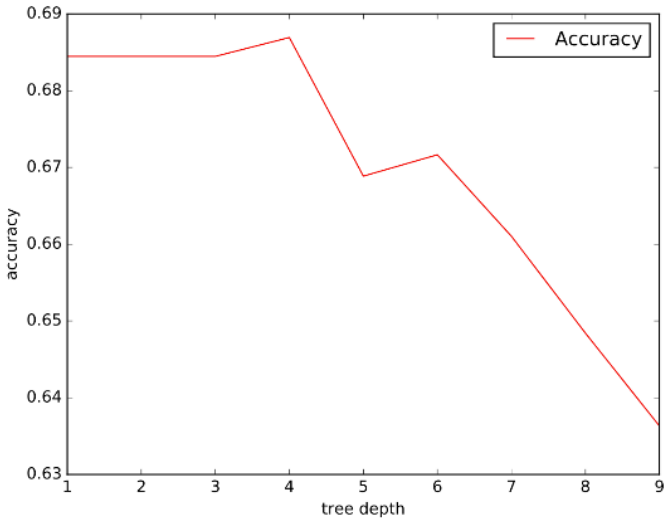

Fig. 4: Performance for different tree depth

Number of iteration of training	10000
Learning rate	0.0001
Number of units of hidden layer1	50
Number of units of hidden layer2	25
Input data	USD/EUR, DAX, Hang Seng, Dow jones industrial average, Nikkei 225
Adam (Adaptive Moment Estimation) parameters	$\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$

TABLE VII: Feed Forward Artificial Neural Network Base Model

2) Changing Types of Input Data: We attempted to change the types of input data and to analyze what changes may have occurred in our predictions. Please note that the other conditions were same as they were in the base model.

All model	USD/CHF, USD/EUR, USD/JPY, USD/GBP, USD/CAD, USD/AUD, NASDAQ Composite, S&P500, Dow jones industrial average, All Ordinaries, DAX, Hang Seng, Nikkei225, Gold and Crude oil
Currency model	USD/CHF, USD/EUR, USD/JPY, USD/GBP, USD/CAD, and USD/AUD
Stock model	NASDAQ Composite, S&P500, Dow jones industrial average, All Ordinaries, DAX, Hang Seng, and Nikkei225

TABLE VIII: Changing types of input data

As for the results, please refer to the figures below. According to the results, both the accuracy and the F1 score of the currency model were inferior to the other models. In other words, foreign currency indices alone were not good predictors for S&P 500 prediction. However, basic, all, and stock models nearly had the same accuracy and F1 score and it revealed that some stock indices were important predictors for the prediction, and the weights that were related to stock indices must be weighted as "high". Accuracy of any models did not reach to 70%, so the accuracy limit for our model with data types and period that we gathered must have been around 70%.

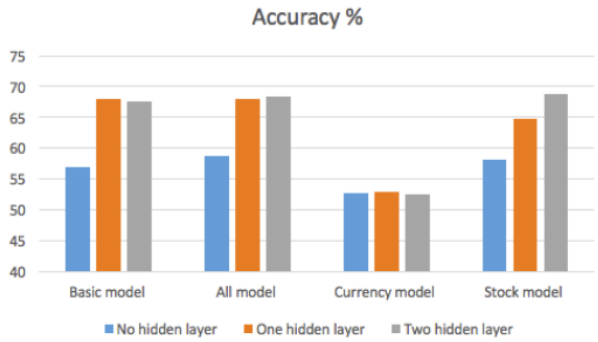

Fig. 5: Accuracy % by types of input data

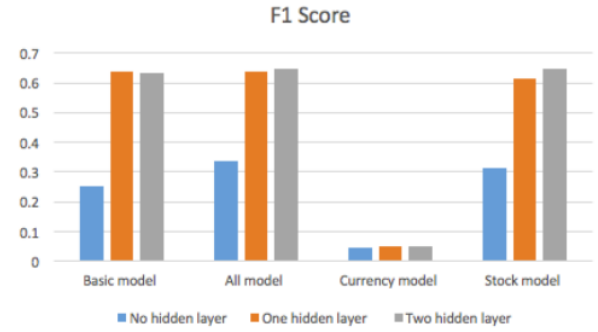

Fig. 6: F1 Score % by types of input data

3) Changing Learning Rate: We attempted to change the learning rate and to analyze what changes occurred in our prediction. Again, the other conditions were same as they are in the base model.

Regarding the one and two hidden layer models, the accuracy and the F1 score were nearly identical and the learning rate seemed to have little influence. Yet, no hidden layer model increased both the accuracy and the F1 score along with the learning rate rising. We assumed that one and two hidden layer models could learn more efficiently than a no hidden layer model and, that a 10000 times learning iteration was enough for such models. On the other hand, if the learning rate was too low, we thought that a no hidden layer model could not learn thoroughly enough or escape local optima even if learning 10000 times, and the correct answer rate to the test data became low. However, if the learning rate was enough high, no hidden layer model could learn enough or escape the local optima, so accuracy was also higher than 60%.

When the learning rate equaled 0.00005, F1 score of the no hidden layer model was extremely low compared to the accuracy. For this reason, we assumed the model tends to predict S&P 500 closes down, even if actual S&P 500 closes up. In other words, the model predicts S&P 500 closes up with low accuracy and S&P 500 closes down with high accuracy. According to the definition of F1 score, it may be the cause of the very low F1 score.

4) Changing Number of Units of Hidden Layers: We also changed the number of units of hidden layers and analyzed the results. Other conditions were same as the base model.

In terms of the one hidden layer models, both the accuracy

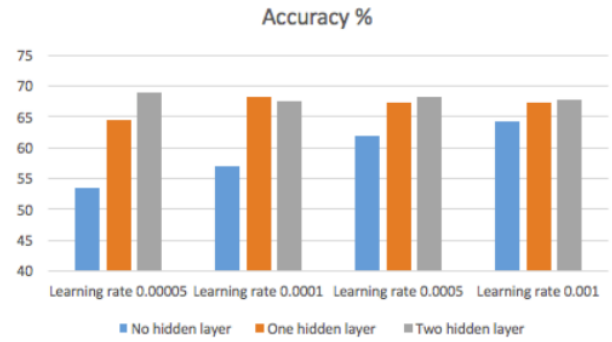

Fig. 7: Accuracy % by learning rate

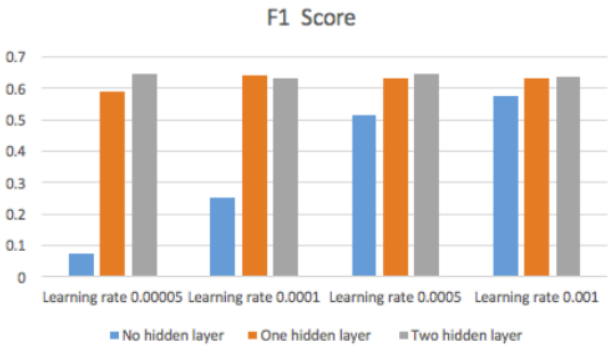

Fig. 8: F1 score by learning rate

and the F1 score increased along with the number of units of hidden layer that rose. It was because the model could not learn well with too few units of hidden layers. Up to 50 units, the accuracy has increased along with the increase in the number of units, but after that the accuracy decreased slightly. This might have been because of "overfitting" due to too many units.

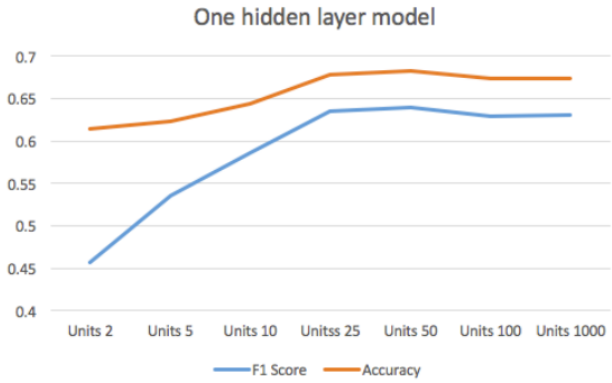

Fig. 9: Accuracy and F1 score of one hidden layer model by changing number of units

In terms of the two hidden layer model, we first fixed the number of units of hidden layer 1 to 50 and changed the number of units of hidden layer 2. As we expected, if the number of units of hidden layer 2 was very small, both Accuracy and F1score were low. However, when the number of units of hidden layer 2 was 5 or more, big change could not be seen.

When the number of units of hidden layer 1 was fixed to 2, the graph became as Figure 11. Accuracy and F1 score of two hidden layer model U1 = 2.

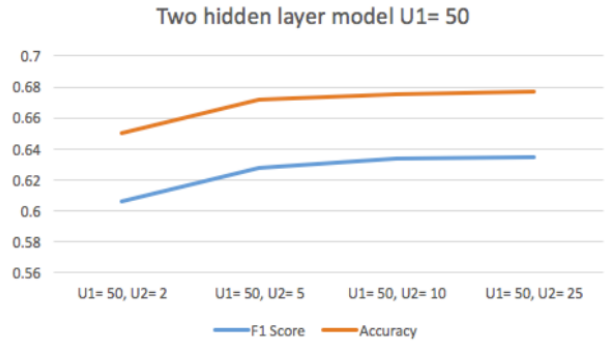

Fig. 10: Accuracy and F1 score of two hidden layer model U1 = 50

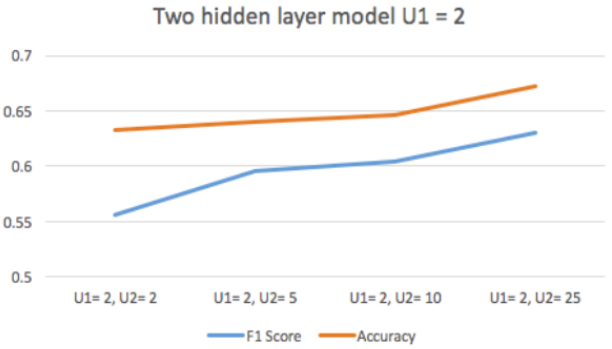

Fig. 11: Accuracy and F1 score of two hidden layer model U1 = 2

Again, as the number of units of hidden layer 2 grew bigger, Accuracy and F1 score were increasing. Also, since U1 was as small as 2, the overall scores were lower than when the number of units of hidden layer 1 was 50. As a result, we reconfirmed that if the number of units of hidden layers is too few, the model cannot learn effectively.

V. Conclusion

As a result, using SVM, decision tree and FFNN models, we were able to predict whether S&P 500 will go up or down for the test data to some extent. TABLE IX shows best accuracy and F1 score of each model. By adjusting some parameters, we could achieve 67% accuracy for the no hidden layer model, but the no hidden layer model's accuracy and F1 score were generally much lower than other models. In terms of other four models, the accuracy for our test data was almost same. However, feed forward neural network models had lower F1 scores than SVM and decision tree models. As we mentioned before, We assumed this was because the models tended to predict S&P 500 will close down and when S&P 500 actually will close down, they can predict with high accuracy, on the other hand, when S&P 500 will close up,

the model often predicts S&P 500 will close down. Due to definition of F1 score, the tendency would harm F1 score.

TYPE	SVM	Decision Tree	Hideen Layer(s)		
			No	One	Two
F1 SCORE	0.72	0.73	0.63	0.64	0.65
ACCURACY	68%	69%	67%	68%	69%

TABLE IX: Comparison of best accuracy and F1 score of our models

The SVM and the Decision Tree models seemed to be superior methods for our S&P 500 closing prediction algorithms. However, it was difficult to say which model was the best to predict S&P 500 because these accuracy and F1 score fluctuate by periods as well as types of training and test data.

We believe that these models must have higher accuracy than human decision that is based on experts' experience and intuition. However, we also feel that it is difficult to make money by using these models in real stock trading. It is because we have to pay transaction fee for securities companies. Moreover, we can predict if today's S&P 500 would close up or down based on past input data before the S&P 500 market opens. However, when the market opens, it is likely that the S&P 500 price has already moved in the direction our model predicts against the close price of yesterday's S&P 500.

Having said that, machine learning methods such as SVM, decision tree and FFNN worked for predicting the stock price to some extent and we saw the potential of the machine learning technologies through the project. We hope to utilize the knowledge we learned from the project in our future careers.

VI. Other Studies

PREDICTING stock prices is important not only for speculators and traders who aim to earn profits in the short term, but also for shareholders and companies holding shares over the long term. It is because if they anticipate a decline in stock prices due to changes in the external environment, they could hedge using future and optional transactions and reduce the decrease in assets.

However, human beings often make misjudgments from the influence of psychological biases and emotions, and often fail in predicting stock prices. Therefore, stock price prediction based on machine learning algorithms by utilizing historical data that is not affected by emotions and biases is highly important, and many researches are undertaken academically. In fact, the algorithmic trading accounts for about 70 percent of American stock market transactions even in 2010 [11].

Academic researches about stock price forecasting by machine learning algorithms are diversifying year by year. Therefore, first of all, we would like to classify by researches' goals.

Like our research, many researchers all over the world tried to predict the direction of the stocks going up or down [12] [13] [14] [15] [16] [17] [18] [19]. It is because fluctuation of stock prices is very complex, nonstationary, and noisy due to various factors. Moreover, prediction of stock prices is more difficult and less accurate than prediction of the direction in which the stock prices will rise or falls [16].

However, many algorithms were developed in order to predict actual stock prices, returns, or performance in specific periods unlike our research [20] [21] [22] [23] [24] [25] [26].

In addition, there were quite a few interesting systems to achieve unique goals. For example, Faynburd A and Luo LChen X tried to develop systems to predict the turning point at which stock prices changed from rising to declining or from declining to rising [27] [28].

Moreover, Wang F, Zhang Y, Rao Q, Li K, and Zhang H tried to predict volatility. They evaluate the systems not only by accuracy but also speed to predict the volatility because the speed to predict volatility is very important in order to earn money by utilizing these systems in the real financial world [29].

There are many approaches to achieve high performance to predict the direction of stock prices, like our research. Yet, support vector machine seems to be the most popular method to predict the direction of stock prices because the support vector machine algorithm was used in all of eight dissertations that we gathered for predicting the directions of stock prices [12] [13] [14] [15] [16] [17] [18] [19].

For example, Kai-Sang Leung C, Kyle MacKinnon R, and Wang Y introduced structural support vector machine (SSVM) model to forecast binary prediction about the stock market prices. In which, the authors tested different values of the cost function for the support vector machine to find the optimal cost parameter. They split their prediction result into two different categories, one for negative class (going down), the other one for positive class. According to their result, for the negative class, their SSVM had about 63% accuracy for their test data, while for the positive class, it comes down to 57% [12]. For our prediction model, we did not separate our prediction results into two different classes and measured the accuracy for whole testing data set. The accuracy of our support vector machine was about 68% that is higher than the accuracy of their SSVM. We believe this is because we also tested other parameters such as different kernel functions, while they only paid attention to the cost function parameter.

Some researcher's conducted comparison between the performance of support vector machine and other algorithms. For example, Kara Y, Acar Boyacioglu M, and Kaan Baykan tried to compare the accuracy of support vector machine with the accuracy of artificial neural network for ISE National 100 Index. As a result, the average accuracy of the ANN model (75.74%) was better than that of the SVM model (71.52%) [13].

Next, Abdullah Zeino Y compared the performance of support vector machine with the decision tree algorithm for GCC Stock Indices. He claimed that the support vector machine is generally superior to the decision tree algorithm, and the performance of the linear kernel outperforms other non-linear kernel [14]. In our experiments, we also found that the linear kernel is superior to other non-linear kernels. Therefore, regarding the type of kernel, the results of our experiments agreed with his claim.

Wang Y analyzed the performance of support vector machine and artificial neural network. The average accuracy of artificial neural network on forecasting indices (KOSPI and

HIS) is generally better than support vector machine. On the other hand, the average accuracy of support vector machine on forecasting individual stocks is generally better than artificial neural network. He also revealed that the volatility of artificial neural network for predicting both indices and individual stocks is higher than the volatility of support vector machine. In other words, artificial neural network has problems about high-volatility and overfitting. If an artificial neural network is over-fitted by training data set, the efficiency of prediction for test data set decreases rapidly. It means that noisy and nonstationary stock price information would lead artificial neural network to a too complex model [15]. Moreover, Cortes, C. and Vapnik, V claimed that support vector machine often achieves better generalization performance and lower risk of overfitting than artificial neural network algorithms because support vector machine implements the structural risk minimization principle [30]. It may be the cause that our neural network models have lower F1 scores than the support vector machine and the support vector machine algorithm seemed to be superior to neural network algorithms.

Finally, Phichhang Ou used various algorithms, such as Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), K-nearest neighbor classification, Naive Bayes based on kernel estimation, logit model, tree based classification, neural network, Bayesian classification with Gaussian process, Support Vector Machine (SVM) and Least Squares Support Vector Machine (LS-SVM). He compared them and concluded that the support vector machine and the least squares support vector machine are generally superior to other algorithms [16].

In conclusion, according to their results, although superiority of algorithms depends on types as well as periods of data set and algorithm parameters, the support vector machine and artificial neural network are certainly reliable algorithms to employ in predicting stock price direction.

The researchers also included various kinds of input data as well as methods to decide the input data set. Especially, technical indicators are widely used in experiments to predict the direction of stock prices [13] [14] [19]. For example, Ni L, Ni Z, and Gao Y used 19 technical indicators and the fractal feature selection method that is suitable for solving the non-linear problem to spot how many important features they should select [19].

On the other hand, Phichhang Ou used not only open, high, and low of Hang Seng index, but also S&P500 and USD/HKD as input data set [16]. Madge S used price volatility and the momentum of individual stocks and the technology sector to predict 34 technology companies' stock prices [17]. Heo J and Yong Yang J uniquely utilized financial statement data, such as earning per share(EPS), and net profit as input data set [18].

We believed that gathering many types of input data set such as stock indices, exchange rate, and commodities is one of the unique points of our project. Even if the performance of our systems is more accurate than other people's systems, it does not imply our systems are more superior than the other systems. It is because we used data on the same date as S&P500 regarding the stock prices of other countries' indices such as Hang Seng index and Nikkei 225. However, other researchers generally used input data one day ago from the target stock price data. Therefore, it is natural that the accuracy of our systems is higher than the experiments that use stock prices from one day ago as input data. As we mentioned, in the real world, we can predict if S & P 500 will go up or down before the market starts, but it is likely that when S&P 500 market opens, S&P 500 price has already moved in the direction that our systems predicted. Therefore, we think it is difficult to make money by utilizing our systems.

REFERENCES

[1] Adam Optimizer, https://www.tensorflow.org/versions/r0.11/api_docs/python/train.html#AdamOptimizer

[2] Decision Tree - What are the?, http://support.sas.com/publishing/pubcat/chaps/57587.pdf

[3] Machine Learning with Financial Time Series Data, https://cloud.google.com/solutions/machine-learning-with-financial-time-series-data

[4] Jason Weston. Support Vector Machine Tutorial. NEC Labs America http://www.cs.columbia.edu/kathy/cs4701/documents/jason_svm_tutorial.pdf

[5] Kingma, D. P., & Ba, J. L. (2015). Adam: a Method for Stochastic Optimization. International Conference on Learning Representations, 1-13

[6] Qaundl Financial and Economic Data, https://www.quandl.com/

[7] Scikit-learn Decision Tree Classifier http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier

[8] Scikit-learn SVC, http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

[9] Scikit-learn Support Vector Machines, http://scikit-learn.org/stable/modules/svm.html

[10] Tensorflow, https://www.tensorflow.org/

[11] Frank Zhang The Effect Of High-Frequency Trading On Stock Volatility And Price Discovery. 1st ed. X. 2010.

[12] Kai-Sang Leung C, Kyle MacKinnon R, Wang Y. A Machine Learning Approach For Stock Price Prediction. 1st ed.; 2014.

[13] Kara Y, Acar Boyacioglu M, Kaan Baykan . Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. 2010.

[14] Abdullah Zeino Y. Forecasting the Direction of GCC Stock Indexes Using Support Vector Machine (SVM) Algorithm. 2014.

[15] Wang Y. Stock price direction prediction by directly using prices data: an empirical study on the KOSPI and HSI. 2014.

[16] Ou P. Prediction of Stock Market Index Movement by Ten Data Mining Techniques. 2009.

[17] Madge S. Predicting Stock Price Direction using Support Vector Machines. 2015.

[18] Heo J, Yong Yang J. Stock Price Prediction Based on Financial Statements Using SVM. 2016.

[19] Ni L, Ni Z, Gao Y. Stock trend prediction based on fractal feature selection and support vector machine. 2010.

[20] Chen R, Pan B. Chinese Stock Index Futures Price Fluctuation Analysis and Prediction Based on Complementary Ensemble Empirical Mode Decomposition. Mathematical Problems in Engineering. 2016;2016:1-13. doi:10.1155/2016/3791504.

[21] Yu H, Liu H. Improved Stock Market Prediction by Combining Support Vector Machine and Empirical Mode Decomposition. 2012.

[22] Hegazy O, S. Soliman O, Abdul Salam M. A Machine Learning Model for Stock Market Prediction. International Journal of Computer Science and Telecommunications. 2013;4(12).

[23] Luo F, Wu J, Yan K. A Novel Nonlinear Combination Model Based on Support Vector Machine for Stock Market Prediction. 2010.

[24] Cai X, Hu S, Lin X. Feature Extraction Using Restricted Boltzmann Machine for Stock Price Prediction. 2012.

[25] Booth A, Gerding E, McGroarty F. Automated trading with performance weighted random forests and seasonality. 2013.

[26] Karathanasopoulos A, Theofilatos K, Sermpinis G, Dunis C, Mitra S, Stasinakis C. Stock market prediction using evolutionary support vector machines: an application to the ASE20 index. The European Journal of Finance. 2015;22(12):1145-1163. doi:10.1080/1351847x.2015.1040167.
[27] Faynburd A. Autoregressive Short-Term Prediction Of Turning Points Using Support Vector Regression. 1st ed. Cornell University Library; 2011. Available at: https://arxiv.org/abs/1209.0127. Accessed December 5, 2016.
[28] Luo LChen X. Integrating piecewise linear representation and weighted support vector machine for stock trading signal prediction. Applied Soft Computing. 2013;13(2):806-816. doi:10.1016/j.asoc.2012.10.026.
[29] Wang F, Zhang Y, Rao Q, Li K, Zhang H. Exploring mutual information-based sentimental analysis with kernel-based extreme learning machine for stock prediction. 2016.
[30] Cortes CVapnik V. Support-vector networks. Machine Learning. 1995;20(3):273-297