2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)
17-20 October, 2021. Melbourne, Australia

2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC) | 978-1-6654-4207-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/SMC52423.2021.9658938

Deep Learning Vs. Machine Learning in Predicting the Future Trend of Stock Market Prices

Alireza Ghasemieh[1], Rasha Kashef [2]

Abstract - **The ability to predict the stock trend is one of the most challenging goals for today's traders. The successful prediction of a stock's future trend could yield significant profit. Various machine learning and deep learning have been introduced in the last decades. However, the trade-off between performance and computational complexity was not addressed. This paper aims to find a well-suited model to predict the stock market price trend, with increment in profit gain in Long and Short trading with comparable prediction performance and computational time. A state-of-art machine and deep learning methods have been investigated along with efficient feature engineering. Experimental results show that Feed Forward Neural Network (FFNN) has the best profitability performance (return) and a reasonable running time, among other tested models.**

Keywords: Stock Trend Prediction, Machine Learning, Deep Learning, Ensemble Modeling, Feature Engineering, Performance Evaluation.

I. INTRODUCTION

Stock prices are volatile and depend on various factors such as company financial statement, news, transaction cost, market trends, popularity, price of alternate companies, sentiments, macroeconomy parameters, market momentum, and some legal factors. The ability to forecasting the stock price is one of the most challenging goals for AI researchers. Using only the stock price history or adding complementary data are different approaches for trend prediction. However, due to the lack of domain knowledge of the stock market, few research studies have been conducted to combine the advanced technical indicators with price history to forecast the market trends. Almost all professional traders use technical indicators to decide on going long, short or hold a position. The decision-making process is highly complex and should be completed in a few minutes by the trader to take advantage of the small window opportunity. Over the past few decades, various machine learning and deep learning models have been developed to forecast the stock market trend. Wang et al. [1] used artificial neural networks (ANN) to predict the stock market price, focusing on the stock market volume. In [2], they used short-term prediction by applying the support vector machine (SVM) to forecast the stock price. Many researchers also applied signal processing techniques and statistical methods to analyze market behaviour. As artificial intelligence methods and strategies have evolved in recent years, many solutions have attempted to use machine learning and deep learning or combined to serve as powerful tools in predicting the future trend of the stock market. Liu et al. [3] proposed a deep learning hybrid model consisting of a convolutional neural network (CNN) and a bidirectional long short-term memory (LSTM) to evaluate the performance of some quantitative strategies in stock markets. [4]. While the researchers proposed innovative solutions, further investigation is required to determine the trade-off between the training cost compared to the prediction accuracy or quality. Our objective in this paper is to determine which model is well-suited for short-term price trend prediction. Therefore, this paper uses the historical stock price and advanced technical indicators to forecast the next day's price trend. Various machine learning and deep learning models have been investigated, including ensemble models. Experimental results show that the FFNN, Ensemble Stacking, Random Forest, SGD have a positive return and the rest have a negative return. However, considering the model training time, ensemble stacking is not a good choice. The rest of the paper is organized as follows: section 2 discussed the existing methodologies and other approaches to predict stock market trends. Section 3 explains the problem that we aim to address in detail. Section 4 presents the training models which are used in this paper. Section 5 discusses the model evaluation method. Section 6 discusses the experimental evaluation, including the preprocessing steps, feature engineering, feature selection, data formatting, class balancing and results.

II. RELATED WORK

Several methods are proposed to predict stock prices trend. In general, these methods can be divided into two categories based on statistical techniques or machine learning. Machine learning-based methods have a much higher ability than statistical methods to detect behaviour patterns in the stock market. For this reason, in reviewing the literature, we will investigate the machine learning-based methods and refrain from examining the statistical techniques. Statistical methods like autoregressive integrated moving average (ARIMA), generalized autoregressive conditional heteroskedasticity (GARCH) volatility [5], and the smooth transition autoregressive model (STAR) [6] are efficient in prediction as long as the time series data is stationary, linearity and normality. However, these assumptions are not satisfied in stock markets. This problem is also present in machine learning methods, and unlike statistical methods, there is no specific mechanism for dealing with non-stationary data. To address this issue, a new hybrid end-to-end approach containing two stages, the Empirical Mode Decomposition and Factorization Machine based Neural Network (EMD2FNN), was proposed to predict the stock market trend [7]. This method integrates empirical mode decomposition (EMD), Factorization Machine (FM), and neural networks. EMD2FNN shows a good accuracy in handling the nonlinearity and the influence among time

[1] Alireza Ghasemieh is an M.Sc. student at the Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, Canada (e-mail: alireza.ghasemieh@ryerson.ca)

[2] Rasha Kashef is Assistant Professor at the Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, Canada (e-mail: rkashef@ryerson.ca)

scales, challenging existing methods. The stock market price depends on the news, significantly impacting the price within 20 minutes [8]. Therefore, considering this source of information in stock trend prediction is very helpful. Radial Basis Function (RBF) Extreme Learning Machine (ELM) has an eye-catching performance using both news and historical price information. This method is faster than BP-NN and has a similar accuracy performance as RBF SVM [9], which is useful when the processing time matter, especially for intra-day trading. Considering the nature of stock price, which is time series, using Recurrent Neural Network (RNN) has become popular in the last few years. Xiong in [10] showed that the map of LAPM over Google trend and economic variables is less than the linear model and autoregressive model. Silva et al. in [11] used RNN and historical price and macroeconomic variables to build a forecasting model. The result showed that RNN follows the actual price more precisely. The usage of RNN is not limited to the stock market. It has an acceptable performance in the cryptocurrency market as well. Sean showed that an LSTM network could predict the bitcoin price with 52% accuracy [12]. The same result is obtained by Gao using LSTM to forecast six US stock market indexes with an accuracy of 54.83% [13]. Pang et al. [14] compared the performance of the LSTM embedded layer and automatic encoder to the vectorized data. The result showed LSTM with embedded layer has a better performance of 57.2%. Besides RNN, some studies have been conducted using other types of deep learning. Yan and Zhao [15] showed a hybrid CNN model, and SVM has an acceptable performance in predicting the American and European stock index. Liang et al. [16] used Restricted Boltzmann Machine (RBM) for feature extraction and incorporated it with SVM, RF, and LR to predict the stock market trend. The raw data is used to generate 11 technical indicators and then are passed through an RBM network, and the result is used to train the predictors. The result showed accuracies of 61.5%, 61.9%, and 60.7% for proposed models, respectively. EAal et al. [17] proposed a trend prediction system based on Artificial Neural Networks (ANN) and fuzzy logic rules. The model is trained over the Egyptian stock market for 3 years. The system has two parallel processing. The first one gets the raw data and performs feature extraction and data preparation to add technical indicators. Then it selects the essential features and finally uses the neural network as a classifier. The second process gets the raw data and uses a Fuzzy Rule-Based Model named Elliot Wave Theory to create trend prediction rules. Both outcomes are integrated into a neural network to produce the final prediction. The model's result obtained an excellent accuracy of 90%, which was way better than MLP, SVM, and RBF classification methods. Another approach to predicting the stock movement is decomposing the price signal, filtering the noisy component with different filtration methods like wavelet [7], and then using the rest to predict the market trend. Wen proposed a new method to filter noisy components from financial data via sequence reconstruction by leveraging motifs and then use CNN to feature extraction. The result showed that the model's accuracy increased by 4% to 7% compared to traditional signal processing methods [18].

III. PROBLEM DEFINITION

Predicting the next day's market trend is a classification problem. Each trade is an independent game that starts in the morning, mainly before the market opening time (pre-bid), and finishes by the end of the day. The bought and sold stock is not allowed to be kept for the next day. Therefore, winning or losing the game is determined at the end of the day. Many games on different stocks are started based on the price trend prediction on each stock during each day. The daily portfolio balance is calculated based on the number of losing and winning. The balance and more related metrics are considered as the classifier performance evaluation metrics. If the classifier forecasts 51% correctly and the number of trades becomes big enough, then the balance would be positive, which means the overall trades profit is more than the loss. Because the traders win two more games pre 100 trials and gain more than what they lose. Of course, if the classifier reaches a higher percentage of correct prediction, it allows more profit. In this problem, winning and losing have the same weight in balance calculation. We should mention that trade fee plays a significant role in the profitability of a model. We will discuss it later.

IV. THE LEARNING MODELS

In this paper, multiple machine learning classifiers, including K-Nearest Neighbors (KNN), Decision Tree, Random Forest, Support Vector Machine (SVM), Naïve Bayes, Stochastic Gradient Descent (SGD), XGBoost, Stacking Ensemble Model, and two deep learning models as Feed Forward Neural Network, and LSTM are adopted.

A. K-Nearest Neighbors (KNN)

The KNN is an instance-based learning algorithm. When a new instance comes, it is compared to existing instances and finds the nearest neighbour using a distance metric like Euclidean distance (1) and assigns the neighbour's class to the new one. Two methods present the training data in a tree structure: *kD-trees* and *ball tree,* which in the current case, kD-tree is used.

$$ED = \sqrt{\left(a_1^{(1)} - a_1^{(2)}\right)^2 + \left(a_2^{(1)} - a_2^{(2)}\right)^2 + \cdots + \left(a_k^{(1)} - a_k^{(2)}\right)^2} \quad (1)$$

- where: $a^{(1)}$and $a^{(2)}$ are two instances with k attributes.

B. Decision Tree

A decision tree is a recursive divide-and-conquer fashion classifier that stores the training in a tree structure. It uses the *information gain ratio* (IGR) (2) to classify an instance.

$$IG(t) = -\sum_{i=1}^{m} p(c_i)\, log(c_i) + p(t) \sum_{i=1}^{m} p(c_i|t) + p(t) \sum_{i=1}^{m} p(c_i|\bar{t}) log p(c_i|\bar{t}) \quad (2)$$

where:

- c_i represents the i^{th} category
- $p(c_i)$ is the probability of the i^{th} category
- $p(t)$ and $p(t)$ are probabilities that the attribute appears or not in class.

$$IGR = \frac{gain(attribute)}{intrinsic\ information(attribute)} \quad (3)$$

C. Random Forest

Random forest is a bagging ensemble learning algorithm that is based on random decision trees. It uses bootstrap to train a group of trees. It increases the model accuracy by reducing the variance (5) by voting (4) (for classifier) or averaging (for numeric prediction). Each learning iteration selects N random number of instances from the training set with replacement, builds a tree, and stores it. After completing the learning process, it predicts the new instance's class based on voting from trees.

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b \quad (4)$$

where:

- $\hat{f}$ is the predicted variable
- B is the total number of trees
- f_b is classification or regression tree

$$\sigma = \sqrt{\frac{\sum_{b-1}^{B}(f_b(x\prime)-\hat{f})^2}{B-1}} \quad (5)$$

- where: σ is the standard deviation of prediction from all trees at the point of $\acute{x}$.

D. Support Vector Machine

Support vector machines use linear models to classify the non-linear dataset. It uses different kernels to find the best map between attributes space and another higher dimensional attribute vector space. Equations (6) and (7) show instances in the linear and non-linear for *n* factors in the transformation, respectively. The non-linear *kernel function* is used to find the weights for non-linear separable datasets.

$$x = w.a = w_0a_0 + w_1a_1 + \cdots + w_ka_k \quad (6)$$

$$x = w.a = b + \sum_i \alpha_i\, y_i\, (a(i).a)^n \quad (7)$$

E. Naïve Bayes

The Naïve Bayes is a supervised learning algorithm based on Bayes' theorem, assuming a "naive" conditional independence between each of the two properties with respect to the variable value of the class.

F. Stochastic Gradient Descent

Stochastic gradient descent (SGD) is an efficient method for adapting linear classifiers and regressors under convex loss functions such as logistic regression.

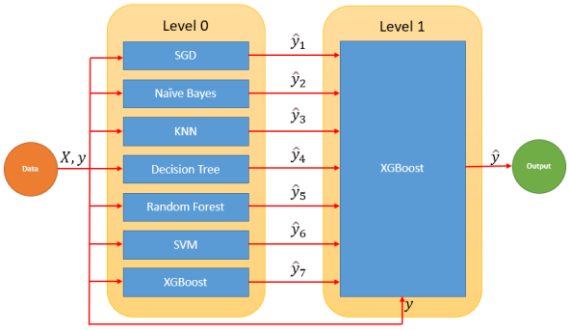
G. XGBoost

The XGBoost is a robust boosting ensemble learning algorithm that uses gradient descent to minimize the error and has a regulation option to avoid overfitting while using many trees. It is designed for speed and performance by Tianqi Chen. This model uses a parallel process during training. It uses the distributed computing and out-of-core computing for enormous datasets. Moreover, it optimizes the cache to provide the best usage of hardware [19].

H.Stacking Ensemble Model

The proposed model in Fig.1 is a stacking ensemble classifier. The preprocessed data, including independent attributes vector, denoted by X and class attribute, denoted by y, are fed into all level-0 models. After the predictions have been made, the outputs of level-0 models denoted by ŷ are fed into the metamodel as inputs, which is an XGBoost. Besides the mentioned inputs, the class attribute is also given to the metamodel to check which classification has made a correct or incorrect prediction. This architecture results in a better performance than the individual models. In this architecture, the level-1 classifier is responsible for weighting the output of the level-0 classifiers and makes the final classification decision.

Figure 1. Stacking Ensemble Classifier Architecture



I. Feedforward Neural Network

The Feedforward neural network is a powerful learning model. It is composed of input, hidden, and output layers. This paper used one hidden layer with 100 neurons with ReLU as the activation function, *Sparse Categorical Crossentropy* as the loss function, and *Adam* as the optimizer method. The hidden layer has a dropout layer with 20% to avoid overfitting. Fig. 2 shows the network scheme with an input of 17 attributes, the dense layer has 100 neurons, and the output has three classes.

J. Long Short-Term Memory (LSTM)

The Long Short-Term Memory is a variant of RNN used for the time series dataset. It overcomes the problem of vanishing gradient in RNN. Fig. 3 shows the network architecture. The input has 17 attributes of 7 days. The LSTM layer has 100 neurons, and the output has three classes. The ReLU is used as the hidden layer activation function. Softmax is the output layer activation function with *Sparse Categorical Crossentropy* as the loss function and *Adam* as the optimizer method. The hidden layer has a dropout layer with 20% to avoid overfitting.

Figure 2. The FFNN Scheme

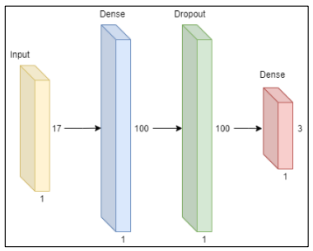
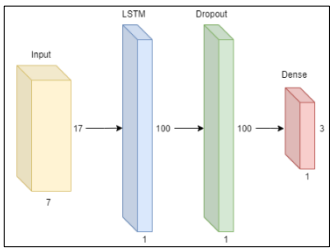


Figure 3. LSTM Network Architecture



V. PERFORMANCE MEASURES

The model evaluation heavily depends on the nature of the problem. The traditional metrics like accuracy, recall, precision, and F1-score are good but do not project the final goal of the modelling. In this paper, we also measure the profitability of the classifiers. The model performance evaluation is based on the *cost matrix* and related ratios. To

understand each model, the traditional evaluation metrics are provided as well.

Cost Matrix

The most important metric to evaluate the model is the Cost Matrix (Table 1). The matrix is designed based on the confusion matrix, where predictions are weighted based on each prediction situation's profit and loss. If the actual class is "Hold" and the classifier predicts up, down, or hold, there is almost no trading loss. Because the stock price did not change. If the actual class is "Up" and the classifier predicts "Hold", the trader just missed the opportunity, but there is no cost. If the classifier predicts "UP", there is 1 profit and predicts "Down", there is 1 unit of loss. If the actual class is "Down" and the classifier predicts "Hold", the trader just missed the opportunity, but there is no cost. If the classifier predicts "UP", there is 1 loss and predicts "Down", there is 1 unit of profit. Table I shows the mentioned logic.

A trading fee (μ) equal to 0.2 unit is considered for all trades, which is used to calculate the final profit. The reason why we chose 0.2 is, each trade has a value of 1000$, and usually, 5$ - 6$ is the trade fee. Since we consider over 3% price change as a trading opportunity, then 30$ is expected to profit. Therefore, if we consider 1 unit for 30$, then for 5$ - 6$ trade fee, 0.2 unit is reasonable.

TABLE I: THE COST MATRIX

		Predicted Class		
		Up	**Hold**	**Down**
Actual Class	**Up**	1 - μ	0	-1- μ
	Hold	- μ	0	- μ
	Down	-1 - μ	0	1 - μ

VI. EXPERIMENTAL EVALUATION

A. The Datasets

The dataset is collected through an API from the Alphavantage servers, one of the stock providers. It contains 83 stocks that start from January 1st, 2010, to November 27th, 2020. The total number of data records is 196142. The data contains:

- Ticker: The stock symbol of the company
- Date: The date of the stock information
- Open Price: The price of the first trade at 9:30 am
- Low Price: The lowest stock price within the day
- High Price: The highest price within the day
- Close Price: The price of the last trade at 4:00 pm
- Volume: The volume of stock trades within the day

B. Class Label Creation

In this paper, the stock price is daily. For classification, three labels have been used:

- Short (Down): The next day price will drop below the predefined threshold of $\mathfrak{f}$
- Long (Up): The next day price will rise above the predefined threshold of $\mathfrak{f}$
- Hold (No Change): The next day price will remain within the predefined threshold $\pm$

If the next day's average price (mean of open and close price) fluctuates above or below $\mathfrak{f}$, it is considered a noticeable price change, and the class label (0: down, 1: hold, 2: up) is built accordingly. The $\mathfrak{f}$ is a hyperparameter that is initiated for $= 3\%$. The result of labelling is:

- The number of uptrend labels: 34k
- The number of downtrend labels: 35k
- The number of hold labels: 100k

C. Data Cleanliness Check

Since NYSE and NASDAQ stock market generates daily stock prices in an elegant format, there is no missing data and outlier in the dataset for being treated at the next steps.

D. Technical Indicators

There is a huge need to use technical indicators to reflect the predictive model's market behaviour and momentum. These indicators are generated using the open, low, high, and close prices and the volume. Many of these indicators use a window of the past days to generate a value to indicate the past trend of the buy/sell of the stock. As a part of the data preparation, we generate some of the most used technical indicators as follows.

a) Commodity Channel Index (CCI)

The Commodity Channel Index (CCI) (14) is a market indicator used to track market movements that may indicate buying or selling. The CCI compares the current price to the average price over a specific period [21]. Different strategies can use the CCI differently, including using it across multiple timeframes to establish dominant trends, pullbacks, or entry points into that trend. Some trading strategies based on CCI can produce multiple false signals or losing trades when conditions turn choppy [21].

$$CCI = \frac{(Typical\ Price - Simple\ Moving\ Average)}{(0.015 * Mean\ Deviation)} \quad (14)$$

b) Moving Average (MA)

The moving average (MA) (15) is used to eliminates the short-term price fluctuation noise and smooth out price trends. MA has various applications in different strategies and even in building new advanced TIs. The most common applications of the MA are to find out the price support and resistance in different time resolutions and the overall market trend. A simple common method is to identify the market trend by checking the cross-over of two different MA like 5-days and 20-days. This method can be used as a trade signal.

$$MA = \frac{A_1 + A_2 + A_3 + \cdots + A_n}{n} \quad (15)$$

- where A is the average price in a period for *n* periods.

c) Average True Range (ATR)

The Average True Range (ATR) (17) is a technical indicator measuring market volatility. It is typically derived from the 20-day moving average of a series of true range indicators. It was initially developed for use in commodities markets but has since been applied to all security types [22].

$$TR = Max\,[(High - Low), Abs\,(High - Close), Abs(Low - Close)] \quad (16)$$

$$ATR = \left(\frac{1}{n}\right)\sum_{i=1}^{n} TR_i \quad (17)$$

- where TR_i is a true range for the *n* time period.

d) Bollinger Band (BB)

Bollinger Bands is a technical analysis tool developed by John Bollinger to generate oversold or overbought signals. It consists of three Bollinger band lines: a simple moving average (middle bar) and up and down bands. The top and bottom bands are usually two standard deviations +/- from the simple moving average of 20 days, but they can be changed [23]. The Bollinger band formula is shown as:

$$Upper\ Band = MA(TP, n) + m * \sigma[TP, n] \quad (18)$$

$$Lower\ Band = MA(TP, n) - m * \sigma[TP, n] \quad (19)$$

$$TP\ (typical\ price) = \frac{(High + Low + Close)}{3} \quad (20)$$

where:

- MA: Moving Average
- n: # days in the smoothing period (typically 20)
- m: Number of standard deviations (typically 2)
- σ[TP, n]: Standard Deviation over last n periods of TP

E. Feature Engineering

A single indicator usually cannot illustrate everything about the future, while combining them can show a promising future trend. The following is the list of the fundamental indicators. Therefore, feature engineering is the next step to define rules, detect those special situations in the market and generate the proper signals. Table II shows newly added features.

TABLE II. NEW FEATURES

Indicator	Description
BB U/L Cross Signal	When the price crosses the Bollinger band upper or lower band, a signal is generated.
5-days MA Slope	The slope of the 5 days moving average line is calculated by the degree.
20-days MA Slope	The slope of the 20 days moving average line is calculated by the degree.
MA_5 & MA_20 Slope Diff	When the 5 days moving average and 20 days moving average cross each other, the slope between them is calculated in degree.
ATR Slope Change	Each day, the ATR line slope is changing, and its difference with the previous day is calculated in degree.
ATR Candle size signal	If the length of the candle size of the day changes more than 1.5 of the previous day, it generates a signal. The strength of the signal depends on the price difference.
CCI signal	If in the last 7 days the CCI moves from -50 to +100 or moves from +100 to -50, two signals are generated.
Price Change	If the day's price changes more than 5%, it generates a signal.

We need to find the co-occurrence of some indicators to detect a significant momentum in the stock market. There is no concrete rule to indicate a buy, hold or sell signal when one or some of the indicators is triggered. Nevertheless, the occurrence itself means there is a massive momentum in the market that provides trading opportunities. Table III shows newly added advanced features.

TABLE III. ADVANCED FEATURES

Combination	Description
Price & Bollinger Band	If the price rises over 3% and crosses the Bollinger upper band, it generates a signal. If the price drops over 3% and crosses the Bollinger lower band, it generates a signal.
Price & ATR	If the price rises over 3% and the ATR candle's size is more than 1.5 of the previous day, it generates a signal., If the price drops over 3% and the ATR candle's size exceeds 1.5 times of the previous day, it signals.
MA Cross & Slope	If the 5-day moving average line crosses the 20-days moving average line and the slope difference is over 30 degrees, it generates a signal.
ATR Slope Change & CCI	Within the last 10 days, if the ATR slope difference is more than 30 degrees, a CCI signal generates a signal.
ATR Slope Change & MA Cross	Within the last 10 days, if the ATR slope difference exceeds 30 degrees and the 5-day moving average crosses the 20-day moving average, it generates a signal.
CCI & MA	Within the last 10 days, if a CCI signal occurs and the 5-day moving average crosses the 20-day moving average, it generates a signal.
5-days & 20-days MA Slope	If the 5-day moving average slope is less than 10 degrees and the 20-days moving average slope is more than -10 degrees, it generates a signal.
ATR Slope Change	If the ATR line slope sign changes either from positive to negative or negative to positive, it generates a signal.

F. Feature Selection

As a crucial step in data preprocessing, the most important features should be selected to be fed into the model. Otherwise, it will destroy the model performance. In this project, two feature selection methods are used, including *Cross-Correlation* and *Backward Elimination*.

a) Cross-Correlation

The independent features that have cross-correlations of more than 0.4 should be identified. Those features with low correlation with the class attribute are removed.

b) Backward Elimination

The backward elimination algorithm checks each feature's relevancy by building a linear model with present and without the presence of a feature and comparing its p-value to find out whether it is essential or not. In this paper, the P-Value threshold is considered as 5%. As a result, 17 out of 30 independent features are selected for training.

G. Data Formatting

The dataset is split into a training and test set with 70% and 30%, respectively, using the time series cross-validator. The data that is fed into an LSTM network should be reformatted to a 3D tensor (21). Since everyday price depends on the previous days, it is a time series problem, and we should consider the past days. The past days' window size (n) is a hyperparameter that is initialized by 7 days. For all other prediction models, each record consists of price and technical indicators. Such that the impact of the previous 20 days is projected in the technical indicators.

$$(\vec{X}, y) = [([\vec{X}_0, \vec{X}_1, \dots, \vec{X}_{n-1}], c_{n-1}), ([\vec{X}_1, \vec{X}_2, \dots, \vec{X}_n], c_n), \dots] \quad (21)$$

- where $\vec{X}_i = [a_0, a_1, \dots, a_{12}]$ and c_j is the class label of instance *j*.

H. Data Classes Balancing

The data set is imbalanced, and the *"Hold"* class has almost three times more data than the *up* and *down* class. The imbalanced dataset causes some issues, primarily when the model performance is evaluated. The down-sampling method is used for all methods except the LSTM to preserve the time series sequence.

I. Results and Discussion

To determine the best model, the profitability of the model (calculated in 22), the total cost of transactions, the number of transactions, and the number of transactions relative to the total number of investment opportunities and the standard deviation from the average of each parameter are considered.

$$Profit = Total\ Win - Total\ Lost - Total\ Trade\ Fee \quad (22)$$

The profitability chart in Fig.4 and Table IV show that the Decision Tree has the highest number of trades and the second-highest loss; because the cost of non-profit transactions has drastically reduced the model's profitability. In contrast, the FFNN model has a much smaller number of trades, but it has a positive profit due to more correct diagnoses than other models. The main reason for the loss in these models is due to the cost imposed on transactions. By considering the trade/opportunity result, which shows the classifiers' profit over the total number of possible opportunities, which is about 1.4k, it is inferable that all classifiers are not good enough to discover the potential opportunities. The maximum number of discovered opportunities belongs to the decision tree with 17%, and the minimum is for SVM with 1%.

Figure 4. Percentage of Profitability

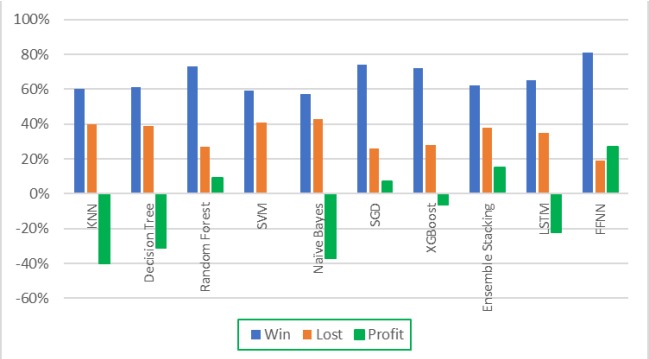


TABLE IV. MODEL PROFITABILITY

Model	Fee	Trade	Trade/Opp	Win/Trade	Lost/Trade	Profit/Trade
KNN	105.0	209.7	16%	60%	40%	-40%
Decision Tree	**120.4**	235.1	17%	61%	39%	-31%
Random Forest	56.6	144.9	9%	73%	27%	**9%**
SVM	9.9	34.1	1%	59%	41%	0%
Naïve Bayes	19.98	36.7	2.5%	57%	43%	-37%
SGD	17.8	54.6	7%	74%	26%	**7%**
XGBoost	45.9	132.1	11%	72%	28%	-6%
Ensemble Stacking	37.4	104.6	8%	62%	38%	**15%**
LSTM	95.0	229.8	14%	65%	35%	-22%
FFNN	30.6	99.5	5%	**81%**	**19%**	**27%**

The FFNN classifier illustrates an incredible Win/Total ratio performance with 81.42%. Thus, the FFNN is capable of predicting the opportunities with high accuracy. However, the classifier total opportunity discovery is shallow, 5%. The FFNN has the highest return with 27%. By enhancing the capability of discovering the opportunities, it will achieve a better return. Fig.5 and Table V show the traditional model evaluation metric: Accuracy, Precision, Recall and F1 Score. Since the 10-fold cross-validation method is used to train and test the model, the boxplots show each model's performance deviation. The accuracy and recall plots show the FFNN has outstanding performance with a good performance on the other two metrics. On the other hance, the Naive Bayes classifier shows an unreliable performance along with all others. However, these metrics are not the final judge for choosing the best model, and profitability is the primary evaluation metric.

TABLE V. MODELS COMPUTATIONAL PERFORMANCE

Model	Accuracy	Recall	Precision	F1_score
KNN	55% (+/-) 11%	55% (+/-)11%	56% (+/-)13%	55% (+/-)12%
Decision Tree	53% (+/-)5%	53% (+/-)5%	53% (+/-)7%	52% (+/-)6%
Random Forest	64% (+/-)7%	64% (+/-)7%	59% (+/-)9%	59% (+/-) 8%
SVM	67% (+/-)7%	67% (+/-)7%	53% (+/-)10%	53% (+/-)8%
Naïve Bayes	40% (+/-)19%	40% (+/-)19%	56% (+/-)13%	36% (+/-)19%
SGD	65% (+/-)7%	65% (+/-)7%	60% (+/-)8%	58% (+/-)8%
XGBoost	61% (+/-)6%	61% (+/-)6%	56% (+/-)8%	57% (+/-)7%
Ensemble Stacking	62% (+/-)5%	62% (+/-)5%	56% (+/-)7%	56% (+/-)6%
LSTM	56% (+/-)8%	56% (+/-)8%	55% (+/-)9%	55% (+/-)8%
FFNN	68% (+/-)6%	68% (+/-)6%	62% (+/-)9%	58% (+/-)7%

The training process using AMD Ryzen 7 1700X Eight-Core Processor took about 17 hours on the complete dataset. The computational training time is shown in Table VI. We can see that the LSTM and Ensemble models have the highest complexity measured by the maximum computational time.

TABLE VI: TRAINING RUN TIME

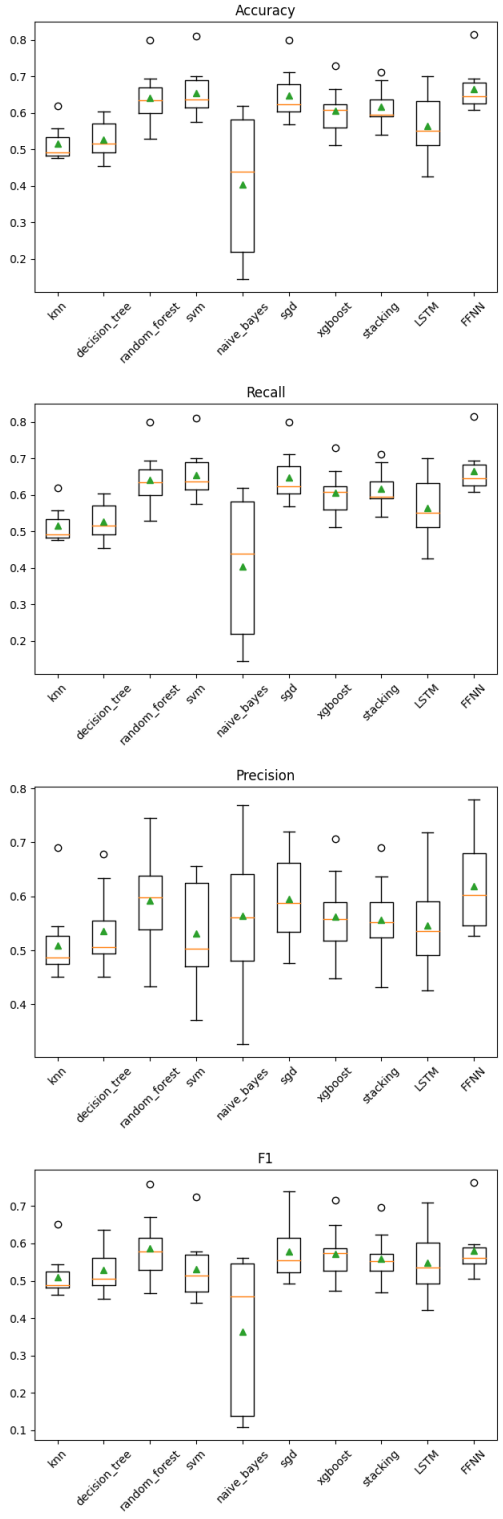
Model	Training Time (Sec)
KNN	30
Decision Tree	30
Random Forest	60
SVM	2400
Naïve Bayes	600
SGD	90
XGBoost	60
Ensemble Stacking	24000
LSTM	33000
FFNN	1200

VII. CONCLUSION

This paper integrates human knowledge with ML using FE and proposes that the FFNN can be considered the best candidate to predict the stock market price trend with high profitability. This prediction offers an excellent opportunity for traders and can help them to facilitate the decision-making

process. Further work is needed to optimize the classifier to discover more opportunities to have a higher trading return.

Figure 5. Performance Metrics



REFERENCES

[1]. Xiaohua Wang, P. K. H. Phua and Weidong Lin, "Stock market prediction using neural networks: Does trading volume help in short-term prediction?," Proceedings of the International Joint Conference on Neural Networks, 2003., Portland, OR, USA, 2003, pp. 2438-2442.

[2]. Ince H, Trafalis TB. Short-term forecasting with support vector machines and application to stock price prediction. Int J Gen Syst. 2008;37:677–87. https://doi.org/10.1080/03081070601068595.

[3]. Liu S, Zhang C, Ma J. CNN-LSTM neural network model for quantitative strategy analysis in stock markets. 2017;1:198–206. https://doi.org/10.1007/978-3-319-70096-0.

[4]. Eapen J, Bein D, Verma A. Novel deep learning model with CNN and bi-directional LSTM for improved stock market index prediction. In: 2019 IEEE 9th annual computing and communication workshop and conference (CCWC). 2019. pp. 264–70. https://doi.org/10.1109/CCWC.2019.8666592.

[5]. P. H. Franses and H. Ghijsels, "Additive outliers, GARCH, and forecasting volatility," Int. J. Forecast., vol. 15, no. 1, pp. 1–9, Feb. 1999, doi: 10.1016/S0169-2070(98)00053-3.

[6]. N. Sarantis, "Nonlinearities, cyclical behaviour and predictability in stock markets: international evidence," Int. J. Forecast., vol. 17, no. 3, pp. 459–482, Jul. 2001, doi: 10.1016/S0169-2070(01)00093-0.

[7]. F. Zhou, H. Zhou, Z. Yang, and L. Yang, "EMD2FNN: A strategy combining empirical mode decomposition and factorization machine based neural network for stock market trend prediction," Expert Syst. Appl., vol. 115, pp. 136–151, Jan. 2019.

[8]. R. P. Schumaker and H. Chen, "A quantitative stock prediction system based on financial news," Inf. Process. Manag., vol. 45, no. 5, pp. 571–583, Sep. 2009, doi: 10.1016/j.ipm.2009.05.001.

[9]. X. Li et al., "Empirical analysis: stock market prediction via extreme learning machine," Neural Comput. Appl., vol. 27, no. 1, pp. 67–78, Jan. 2016, doi: 10.1007/s00521-014-1550-z.

[10]. R. Xiong, E. P. Nichols, and Y. Shen, "Deep Learning Stock Volatility with Google Domestic Trends," Dec. 2015, Accessed: March 21st, 2021. [Online]. Available: http://arxiv.org/abs/1512.04916.

[11]. I. N. da Silva, D. Hernane Spatti, R. Andrade Flauzino, L. H. B. Liboni, and S. F. dos Reis Alves, "Forecast of Stock Market Trends Using Recurrent Networks," in Artificial Neural Networks, Cham: Springer International Publishing, 2017, pp. 221–227.

[12]. S. McNally, J. Roche, and S. Caton, "Predicting the Price of Bitcoin Using Machine Learning," in 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), Mar. 2018, pp. 339–343.

[13]. Q. Gao, Z. He, and T. Supervisor, "Stock market Forecasting using Recurrent Neural Network," 2016.

[14]. X. Pang, Y. Zhou, P. Wang, W. Lin, and V. Chang, "An innovative neural network approach for stock market prediction," J. Supercomput., vol. 76, no. 3, pp. 2098–2118, Mar. 2020.

[15]. X. Yan and J. Zhao, "Application of Improved Convolution Neural Network in Financial Forecasting," in 2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), Apr. 2019, pp. 321–326.

[16]. Q. Liang, W. Rong, J. Zhang, J. Liu, and Z. Xiong, "Restricted Boltzmann machine based stock market trend prediction," in 2017 International Joint Conference on Neural Networks (IJCNN), May 2017, vol. 2017-May, pp. 1380–1387.

[17]. M. M. A. ElAal, G. Selim, and W. Fakhr, "Stock Market Trend Prediction Model for the Egyptian Stock Market Using Neural Networks and Fuzzy Logic," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 6840 LNBI, Springer, Berlin, Heidelberg, 2012, pp. 85–90.

[18]. M. Wen, P. Li, L. Zhang, and Y. Chen, "Stock Market Trend Prediction Using High-Order Information of Time Series," IEEE Access, vol. 7, pp. 28299–28308, 2019.

[19]. T. Chen, "Story and Lessons Behind the Evolution of XGBoost nttrungmt-wiki." https://sites.google.com/site/nttrungmtwiki/home/it/data-science---python/xgboost/story-and-lessons-behind-the-evolution-of-xgboost (accessed March 31st, 2021).

[20]. S. Yan, "Understanding LSTM and its diagrams | by Shi Yan | ML Review | Medium," medium.com, 2016. https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714 (accessed March 21st, 2021).

[21]. K. Wood, Trade the Patterns: The Revolutionary Way of Trading the CCI - Ken Wood - Google Books, Illustrate. W & A Pub, 2009.

[22]. A. HAYES, "Average True Range (ATR) Definition," Investopedia, 2021. https://www.investopedia.com/terms/a/atr.asp (accessed March 21st, 2021).

[23]. A. HAYES, "Bollinger Band® Definition," Investopedia, 2020. https://www.investopedia.com/terms/b/bollingerbands.asp (accessed March 21st, 2021)