JOURNAL OF EMERGING INVESTIGATORS

Comparison of Different Approaches for Stock Price Prediction

David Ahn[1], Odysseas Drosis[2]

[1] Korea International School, Yongin-Si, Gyeonggi-do, South Korea
[2] Computer Science, EPFL, Lausanne, Switzerland

Student Authors
David Ahn, High school

SUMMARY

All people want to earn money, as money is a valuable tool in the modern days. Stocks are a very efficient and profitable method to earn money, as with good timing, minimum loss and maximum profit is possible. However, price change always happens in the future, which humans cannot know. Still, people wanted to predict, which created stock prediction processes using Machine Learning (ML). Out of many ML models, the hypothesis was that the combination of Neural Network (NN) and Linear Regression (LR) model would result in a prediction value below five percent error because the models determine the weights for each value based on past performances, giving chances to improve every prediction trial. To prove the hypothesis, stock prices of Tesla, Apple, and Papa Johns during the past five years were used to train each LR and NN model. Then the test data is used to create a prediction value for each LR and NN, which is compared to real stock price to accumulate the error of each prediction trial. Then, weights for NN and LR are created based on the error ratio, which is used to create a final prediction value. The final prediction value is calculated by adding the multiplied value of the LR weights and prediction value created by the LR model and the multiplied value of the NN weights and prediction value created by the NN model. In conclusion, the hypothesis was correct, since the final average error percentage was 1.97%.

INTRODUCTION

Most people want to earn money in an easy way, but how? Most say that stocks are the answer to that question. Stocks have many definitions, but there is one specific definition related to money and business. It is that stocks are a part of the ownership of a company that can be bought by the members of the public. The stocks' prices vary by the company, as if the company is small, the stock prices is low, and if the company is big, the stock prices are high. The prices change every day based on multiple factors, such as inflation, company activity, interest rate, major investors, consumer spending, and much more [1]. However, it does not mean that investing in the stock market is a great risk. There surely are benefits for investing in the stock market and various reasons of investors for buying stocks, such as capital appreciation, which is the value of stocks rising [2], ability to influence the company's decision, and dividend payment, which is sharing the profit of company to stock owners [3] along with the risks.

However, it is human nature for people not wanting to invest in the unclear future. Therefore, people began to try to predict the stock market change that will happen tomorrow. Stock price prediction is important because in the personal perspective, the prediction tells the future stock price of a company, which the person can use to decide when to bid for the stock and sell the stock. Using these prediction values, an individual can maximize their profit. In the national perspective, if stock values are predicted to suddenly fall like the stock market crash in 1929, then either the government or other organizations can help to counter that prediction to keep the economy as normal.

There are many methods for stock price prediction, but stock price prediction using machine learning (ML) is the trend in the modern days.

Machine learning is a sector of artificial intelligence (AI) that focuses on using data and algorithms to improve accuracy on whatever task it is programmed to do as it imitates the humans' learning method. There are multiple models for ML, but people usually focus on the three types: supervised learning, unsupervised learning, reinforcement learning. Supervised learning is a type of ML that use labeled data to categorize the data or predict outcomes [4]. Unsupervised learning uses unlabeled data and finds patterns in data on its own to cluster data based on characteristics, find relationships between data, or reduce data size since the initial data is too large in dimensions [4]. For the third type, the reinforcement learning machine itself attempts to achieve its mission without prior training of the model to find solutions on its own and maximize profits [5].

Stock price prediction is a popular topic, as many researchers show interest in it. Anshuman and Ayes showed the benefits and the disadvantages that the stock price prediction system can bring [6]. Other researcher even merged five different ML models and used a Root Mean Squared Error (RMSE) method to measure the performance of the merged model [7]. Some researchers even fix the type of company they want to investigate and the time period of the data to correctly measure the performance of currently existing ML model [8]. Indronil and Pyronti even compared the ML approach and the traditional stock prediction approach to see which approach works better to predict stocks [9]. Additionally, other researchers attempt to use the public ML model such as BERT and use its function to and apply to a different type of ML model [10].

Linear Regression (LR) is one of the supervised ML type models. With the data the user uses, which is shown by letter “p”, the weights, shown by letter “w”, are distributed to each of the data.

Then all the data is formed as the equation $w1p1 + w2p2 = p3$. After, the data is divided into the test set and the training set. Using this equation, during the training, the model tries to find the weight values to meet the equation, but there is barely any case when the weight values perfectly fit. Therefore, there are multiple possible weight values created that very slightly do not fit into the equation, which form a slope of best fit or a trend. Then, the weight values that are created through training are tested into the test data in equation form that do not have any exposure to weight values. After, the model finds accuracy based on how much weight values fit into the equation and finds how much weight values are far from the slope, which then gets rid of the negative values.

Neural Network (NN) is also one of the supervised ML type models. After dividing the given data into the train group and the test group, the train group's data is fed into the model, creating an input layer. Then as the data in the input layer moves to the hidden layer, it goes through weights that are assigned randomly by multiplying the data and the assigned weight. It goes through the process of going through the hidden layer multiple times, but now the weights are assigned based on each value's performances, meaning if the value in the hidden layer is close to target value, the model assigns big weights and if the value is far away, the model assigns small weights. After going through all the hidden layers, the values are all added up, which results in the output value. After, the model finds the accuracy based on how much the output value is close to the target value.

Both ML models can be used to predict stock prices of the future for many different companies if they are given the correct and enough data.

Both ML models have advantages that are used for prediction. The NN model can learn and re-evaluate weights based on performance. The LR model can adapt to most of the relationships of data, showing the flexibility of the model. However, there are also negative features. The LR model has a linearity, meaning that the predicted value mostly follows the trend, as the weight values are not modified as each data is only gone through once. Therefore, wouldn't the combination of those two models fix the LR model problem and create a model that amplifies the positive features and reduce the negative features of both models? This experiment had a success chance and was hypothesized that the combination of Neural Network (NN) and Linear Regression (LR) model values would result in a predicted price below five percent error from the real stock price. The error was found out to be 1.97%, resulting in a success in the experiment and a correct hypothesis. This method could also be used in other areas where prediction is needed such as a weather forecast.

RESULTS

This experiment was conducted to see if the combination of the NN model and the LR model can counter each model's negative features and amplify the positive features, as characteristics of two models clearly showed a possibility for success. Additionally, the experiment also included to see which model showed better performance. The error percentage had to be lower than five percent to assume that the hypothesis was correct and showed great performance in stock prediction using each model's positive features.

The data, which is stock prices of Tesla, Apple, and Papa Johns for the past five years, was inputted to each of the variables and was split into train and test data. Afterwards, the LR model and the NN model each used those data to be trained and make predictions for future stock price for each company. Each time the models went through a trial, its errors were accumulated and compared with that of each other model to assign weights for the next trial. After, the weights assigned and predicted price for each model was multiplied, resulting in a final prediction.

The comparison between two models showed that the NN model showed better prediction prices than the LR model (Figure 1, Figure 2). The blue dots show the real stock prices, and the red dots show the predicted stock price by that ML model, and if the two dots seem to overlap, it means that the predicted price and actual price is the same or very similar. The NN model (Figure 2) generally has red dots closer to blue dots than the LR model (Figure 1), showing that the price predicted by the NN model was closer to the real price than the LR model.

The final error ratio between the LR model and NN model was 0.518 to 0.482, also showing the NN model outperformed the LR model. Additionally, the final prediction error percentage was 1.97%, which is below five percent, showing that the combination of two models countered each model's negative features and amplified the positive features.

DISCUSSION

This experiment focused on two objectives: finding out whether the LR model outperformed the NN model or vice-versa, and determining whether the combination of the LR and NN model is effective for stock price prediction.

Some possible limitations with this experiment are that the NN model could not be fed with much data, as even though the time period used for the data was past five years, the number of companies were limited, as there was a limit with human stamina for searching and applying the model. Another limitation might be that the data is not diverse enough to cover all areas of production such as robotics, chemistry, biology, sports, airplanes, and more because the companies used for the data does not focus on many areas, it only focuses on automobiles, electronics, and cuisine. If there were more diverse areas and a greater number of companies used for the data for the models, then the diversity of area the model covers should significantly increase and make sure that the NN model is fed with enough data.

Stock prediction model can further be used to predict the overall status of the economy, as stock prices are not just simply for money, but also is a record for the economic status, as it shows the cycle between demand and supply of the community. Additionally in the future, predicting stock price for a certain date or a period could also be a possible experiment. One remaining question is what will happen if all the ML models that are used for prediction or regression models are combined? Will the combined model be able to perfectly predict the future with enough data? Future experiments can base on these questions.

As result of the experiment, the NN model outperformed the LR model, as the error ratio of those two models were 0.482 to 0.518, meaning the LR model had about 3% more error than the NN model. Adding on, the final prediction model, which was a combination of the LR model and the NN model, had a final average error percentage of 1.97%, which is lower than five percent, the boundary set for the hypothesis to be correct.

The error system used for this experiment was a combination of errors received from the LR model and the NN model. After one single stock price is generated from the data, the models each create a predicted stock price that is similar to the actual stock price. The different between the actual stock price and the predicted stock price are errors, which are used to create ratios that sum up to one, which become the weights for next trial. On the next trial, with different generated actual stock price and created predicted stock prices for each model, the errors are summed up, causing the weights for next trial to change. This process repeats until the last data value and conclude with a final error ratio and a final average error percentage.

With this result, the stock price prediction using a combination of NN and LR model was very successful, and it has a high accuracy enough for people to trust and use it in real life. With this experiment result proved to be practical in real life, many people could attempt to combine other ML models than NN and LR based on their different prediction cases.

MATERIALS AND METHODS

To obtain the data, the yfinance package was pip installed in the google colaboratory. Along with the yfinance package, pandas library was imported as pd, NumPy library was imported as np, and matplotlib library for plotting was also imported as plt.

The actual stock price data was obtained using .Ticker() function for three companies: Tesla, Apple, and Papa Johns. The time period for the data frame was past five years. From the imported data, “High” column, “low” column, “close” column, and “volume” column was dropped from the data, leaving only the “open” column, meaning only the stock prices when the stock market was opened were used as data, which was translated to NumPy array later.

The data was inputted into the matrix with 1250 rows and columns that can either be added or subtracted depending on how many stock values would be used for the prediction, which is the X variable. The predicted stock prices would go into the Y variable.

Now we have the data transferred, preprocessed, and inputted. However, the data still must be split into the train and test data. From scikit-learn, the train_test_split() function was imported and was used to define X_train, X_test, y_train, and y_test variables. As parameters, the pre-defined X and Y variable would be used and the test size was 0.33, which is 33% of the entire data.

The LR model and the NN model were constructed as the data was prepared. From scikit-learn, the linear model was inputted, and the regression() function was used to build the LR model. After, the model was trained using fit() function and was tested. For the NN model, the MLPRegressor() function was inputted with hidden layer sizes 100, which was run fifty times. This NN model was also trained using fit, and the entire model ran 500 times.

Now the models were properly trained, each model's performance had to be compared with each other. As every trial passed, the difference between the predicted stock price and the actual stock price was accumulated as errors. Using those errors, the weight values for the next trial was determined. Additionally, those weight values for each model were multiplied to the predicted stock price to create a final predicted stock price. The errors were also accumulated for the final price, too, and showed a final average error percentage at the end.

REFERENCES

Egan, John. "How Are Stock Prices Determined: The Factors That Affect Share Prices of Listed Companies | Time Stamped." *Time.* time.com/personal-finance/article/how-are-stock-prices-determined/. Accessed 18 Jul. 2023.

Chen, James. "Capital Appreciation: Meaning, Types, and Examples." *Investopedia.* www.investopedia.com/terms/c/capitalappreciation.asp. Accessed 18 Jul. 2023.

"Dividend Payment." *Cambridge Dictionary*. dictionary.cambridge.org/dictionary/english/dividend-payment. Accessed 18 Jul. 2023.

"Supervised vs. Unsupervised Learning: What's the difference?" *IBM.* www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning. Accessed 18 Jul. 2023.

"What is Machine Learning?" *IBM.* www.ibm.com/topics/machine-learning. Accessed 18 Jul. 2023.

Behera, Anshuman and Chinmay, Ayes. "Stock Price Prediction using Machine Learning." *2022 International Conference on Machine Learning, Computer Systems and Security (MLCSS),* 2022, pp. 3-5. doi:10.1109/MLCSS57186.2022.00009

Zhao, Xinyue. "The Prediction of Apple Inc. Stock Price with Machine Learning Models." *2021 3rd International Conference on Applied Machine Learning (ICAML),* 2021, pp. 222-225. doi:10.1109/ICAML54311.2021.00054

Hirey, Manav, et al. “Analysis of Stock Price Prediction using Machine Learning Algorithms.” *2022 International Conference for Advancement in Technology (ICONAT),* 2022, pp. 1-4. doi:10.1109/ICONAT53423.2022.9725888

Bhattacharjee, Indronil, et al. “Stock Price Prediction: A Comparative Study between Traditional Statistical Approach and Machine Learning Approach.” *2019 4th International Conference on Electrical Information and Communication Technology (EICT),* 2019, pp. 1-6. doi:10.1109/EICT48899.2019.9068850

Weng, Xiaojian, et al. “Stock Price Prediction Based On Lstm And Bert.” *2022 International Conference on Machine Learning and Cybernetics (ICMLC),* 2022, pp. 12-17. doi:10.1109/ICMLC56445.2022.9941293

Figures and Figure Captions

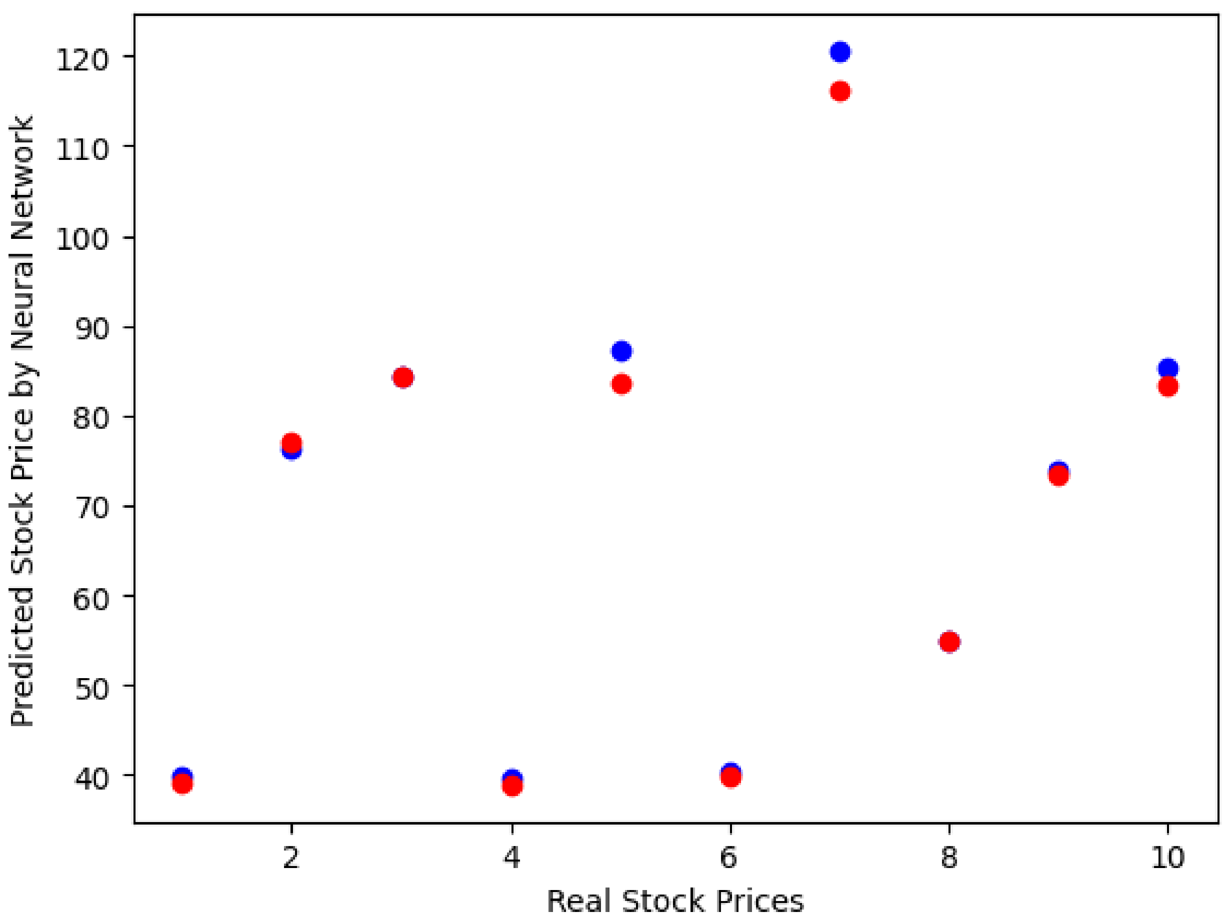

Figure 1. Comparison between real stock prices and predicted stock prices by Neural Network, the red dots are the real stock prices and the blue dots are predicted stock prices generated by Neural Network model, meaning if the two dots are closer, then their prices are more similar. The real stock prices were inputted using tickers of companies.

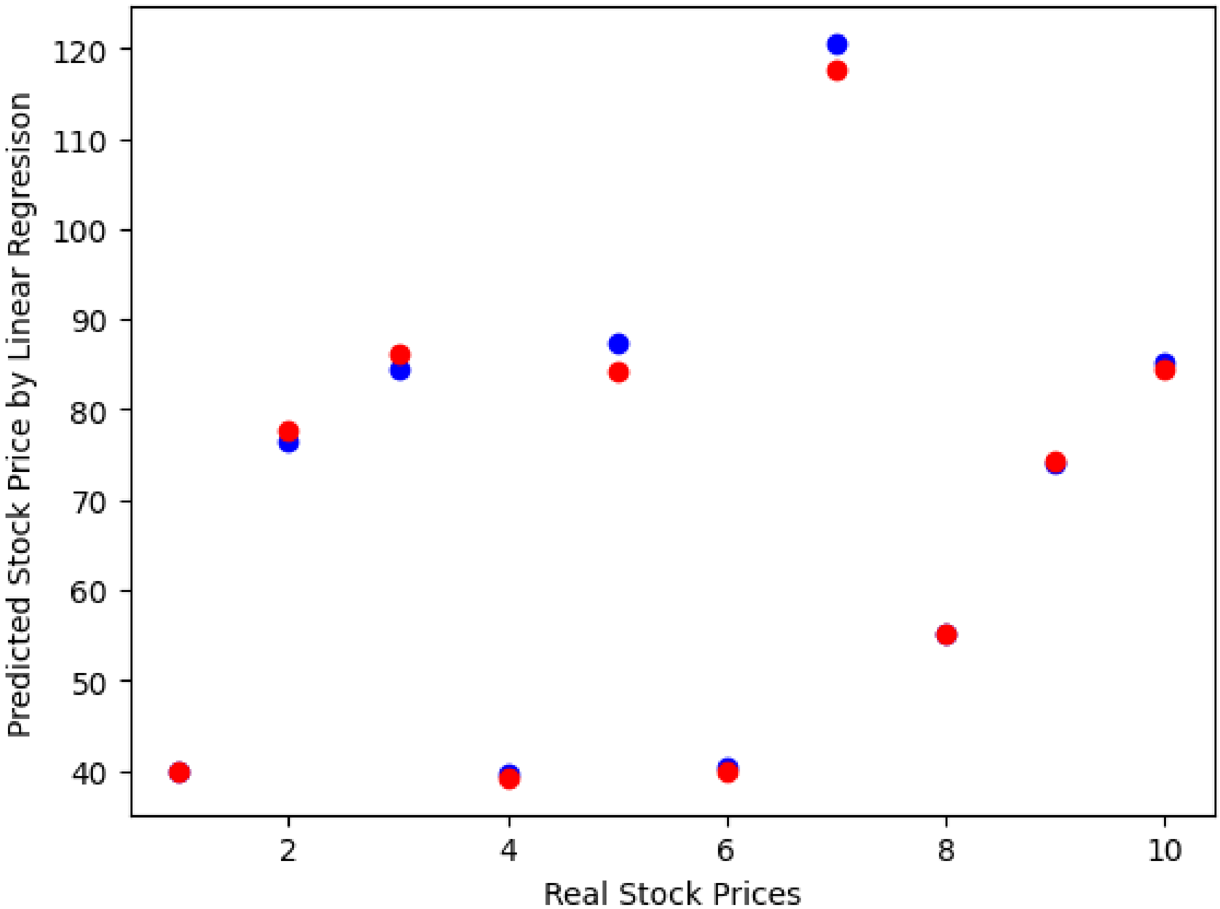

Figure 2. Comparison between real stock prices and predicted stock prices by Neural Network, the red dots are the real stock prices and the blue dots are predicted stock prices generated by Linear Regression model, meaning if the two dots are closer, then their prices are more similar. The real stock prices were inputted using tickers of companies.

Appendix (If applicable)

```
#Installing yfinance package
pip install yfinance
```

```
#Data import

import yfinance as yf
import pandas as pd
import numpy as np

ticker = yf.Ticker('') #import data using the ticker of company you want.
aapl_df = ticker.history(period="") #get data from your selection of timer period.
aapl_df.drop(['High','Low','Close','Volume'], axis=1, inplace=True)
data = np.empty(shape = (1259), dtype = float)
data=aapl_df[['Open']].to_numpy()

X = np.zeros((1250,5))
Y = [0]*1250

for i in range(1250):
  X[i] = [data[1 + i], data[2 + i], data[3 + i], data[4+i], data[5+i]]
  Y[i] = data[6 + i]

from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score

#Split data to test and train sets
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.33)



#Linear Regression Model Setup
LR_regr = linear_model.LinearRegression()

# Training Linear Regression Model
LR_regr.fit(X_train, y_train)

# Make predictions
LR_y_pred = LR_regr.predict(X_test)

print("Coefficients: \n", LR_regr.coef_)
print("Mean squared error: %.2f" % mean_squared_error(y_test, LR_y_pred))
print("Coefficient of determination: %.2f" % r2_score(y_test, LR_y_pred))
```

```python
#Neural Network Model Setup

from sklearn.neural_network import MLPRegressor

#Neural Network Model Train
NN_regr = MLPRegressor(hidden_layer_sizes=(100, 50), random_state=1,
max_iter=500).fit(X_train, y_train)
NN_y_pred = NN_regr.predict(X_test)
print("Mean squared error: %.2f" % mean_squared_error(y_test, NN_y_pred))



#Evaluation using Errors
NN_err = 0
LR_err = 0
fin_err = 0

for i in range(len(LR_y_pred)):
  print("NN Predictions:" , NN_y_pred[i])
  print("LR Predictions:" , LR_y_pred[i])
  print("Actual:",  y_test[i])
  LR_err += abs((y_test[i]- LR_y_pred[i]))
  NN_err += abs((y_test[i]- NN_y_pred[i]))
  LR_ratio = NN_err/(NN_err + LR_err)
  NN_ratio = LR_err/(NN_err + LR_err)
  print("Error ratio is" , LR_ratio, "(LR) :", NN_ratio, "(NN)")
  final_pred = LR_ratio * LR_y_pred[i] + NN_ratio * NN_y_pred[i]
  print("Final prediction value:" , final_pred)
  fin_err += abs((y_test[i]-final_pred)/y_test[i])
  print("")

print("Final Average Error is: " , (fin_err/len(LR_y_pred)) * 100, "%")


#Plotting Neural Network Model Comparison Results
import matplotlib.pyplot as plt
NN_x_values = range(1, len(y_test) + 1)
plt.scatter(NN_x_values[:10], y_test[:10], c='blue', label='Real Stock
Prices')
plt.scatter(NN_x_values[:10], NN_y_pred[:10], c='red', label='Predicted
Stock price by Neural Network')
plt.xlabel('Real Stock Prices')
plt.ylabel('Predicted Stock Price by Neural Network')
plt.show()
```

```
#Plotting Linear Regression Model Comparison Results
import matplotlib.pyplot as plt
LR_x_values = range(1, len(y_test) + 1)
plt.scatter(LR_x_values[:10], y_test[:10], c='blue', label='Real Stock Prices')
plt.scatter(LR_x_values[:10], LR_y_pred[:10], c='red', label='Predicted Stock price by Linear Regresison')
plt.xlabel('Real Stock Prices')
plt.ylabel('Predicted Stock Price by Linear Regresison')
plt.show()
```