CPT_S 570 - MACHINE LEARNING - PROJECT REPORT

TIME-SERIES FORECASTING AND ANALYSIS FOR THE STOCK PRICES

Pallavi Arivukkarasu and Ramyasai Sanjita Bhavirisetty

Department of EECS

Washington State University

ABSTRACT

In the trading environment, high-quality one-step forecasting is usually of great concern to market makers for risk assessment and management. We aim to forecast the price movement of individual stocks based only on their historical price information using 1-dimensional Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM).

1. INTRODUCTION

Predicting the trend has been an ancient discipline, yet it has never fallen from popularity. Whether it is the stock price in the financial market, power or energy consumption, or sales projection for corporate planning, a series of time-based data points can represent how the world is thinking at any given moment, which has always fascinated researchers. The capability to see and react ahead of time is essential in succeeding in many life aspects. It is recorded that 56% of the adults in the United States own some form of stock. Although the number is enormous, the actual number of people who profit from the stock market is minimal. Data analysis is necessary to get the win from the stock market. For example, we can study the movement tendency of the stock price in the last few weeks and then determine whether it will be up or down in the next few days according to some relations between each day’s stock data.

Stock prediction is a very nice idea in one way, but because of its complexity and dynamic features, the prediction result is not satisfied. Some scientists still doubt whether we can make the prediction. In these years, the primary method to predict the stock price movement can be concluded in 2 aspects: machine learning or deep learning. Machine learning is a valuable tool for working in financial areas, divided into supervised and unsupervised machine learning. The significant difference between the former and the latter is whether we have the label of the training data. The core ideology of machine learning is that we should find a supervised mathematical model to fit the data, and then we should train the model to lower the error or improve the fitting degree. Traditional machine learning methods or concepts that people use to help them daily include support vector machines, decision trees, or random forests. Of course, we also can use some mathematical models to help us do the analysis. For example, we can use regression analysis to predict the price of the house. We also can use logistic regression to detect credit card fraud.

2. DATASET AND SOFTWARE USED

Dataset:
Name: Amazon Stock Price 1997 to 2020

Description:
Amazon.com, Inc. is an American multinational technology company based in Seattle that focuses on e-commerce, cloud computing, digital streaming, and artificial intelligence. It is considered one of the Big Four technology companies, including Google, Apple, and Facebook. The dataset contains:

- Date - in format: yy-mm-dd
- Open - the price of the stock at market open
- High - Highest price reached in the day
- Low - Lowest price reached in the day
- Close - The stock closing at the end of the Market hours
- Adj Close - This is the closing price after adjustments for all applicable splits and dividend distributions.
- Volume - Number of shares traded

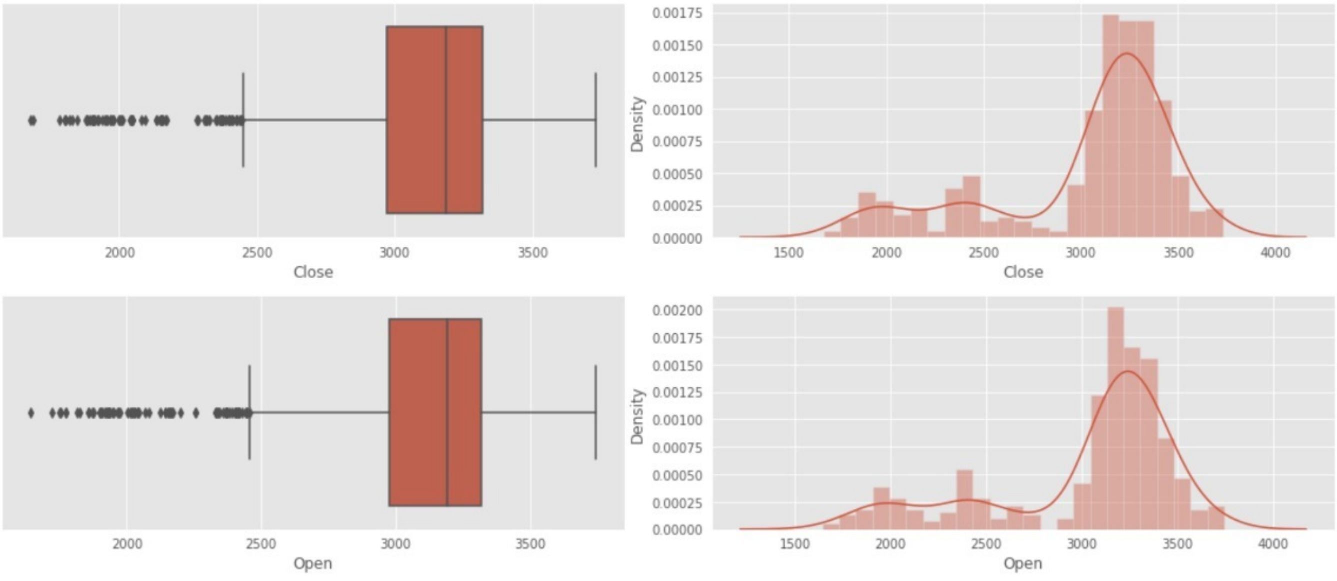
Software/Libraries Used:
Jupyter Notebook, Pandas/Numpy, Matplotlib, Scikit, Keras, Ggplot

3. DATA PRE-PROCESSING

3.1. Cleaning the Dataset

Data preprocessing is crucial when getting data sets to help with prediction. The initial data may have much noise, so reducing them to not interfere with the result is necessary. Besides, because some data features may make no sense, we should neglect them when we train the data to improve efficiency. We got the historical stock data sets of Amazon, which covered from 1997 to 2021 every trading day from the Internet.

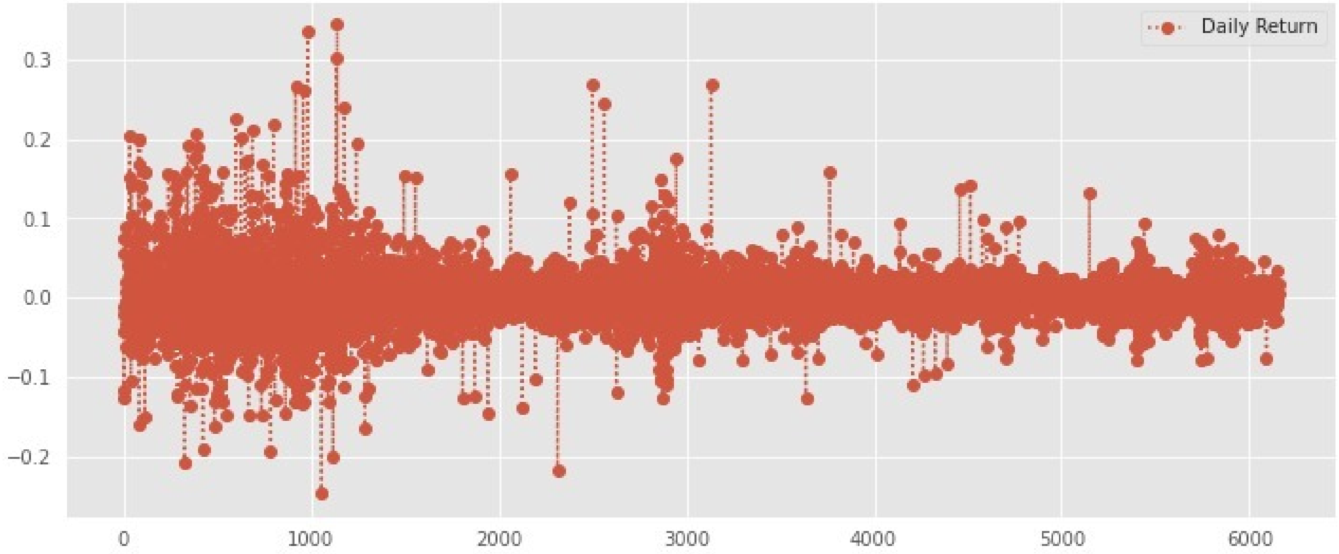
The data initially contains several features, including date, opening stock price, highest stock price, lowest stock price, closing stock price, closing stock price after adjustments, and the volume of shares. We searched for null or missing values in the dataset and removed them, resulting in a cleaner dataset. The mean value has replaced the NA values in the dataset.



3.2. Exploratory Data Analysis (EDA)

Data scientists use exploratory data analysis (EDA) to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods. It helps determine how best to manipulate data sources to get the answers needed, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions.

EDA is primarily used to see what data can reveal beyond the formal modeling or hypothesis testing task and provides a better understanding of data set variables and the relationships between them. It can also help determine if the statistical techniques we consider appropriate for data analysis. We performed EDA to explore and make some analysis with this dataset.



The above figure shows the daily return of the stock prices. It is the percentage of change between the closing prices.

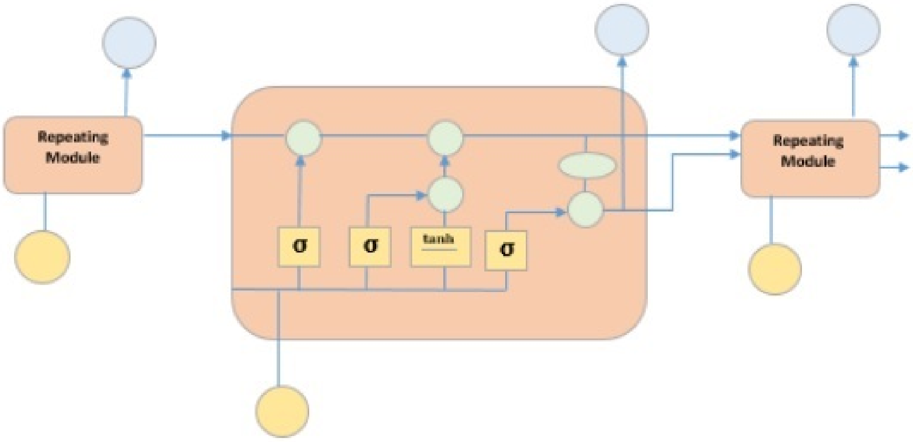
4. METHODOLOGY

4.1. Convolutional Neural Networks (CNN)

Convolutional Neural Network is a feed-forward neural network. Like the traditional architecture of a neural network, including input, hidden, and output layers, the convolutional neural network also contains these features. The output of the convolution layer is the output of the previous convolution layer or pooling. Of course, they still have some unique features, such as pooling layers and total connection layers. The number of hidden layers in a convolutional neural network is more than that in a traditional neural network, which, to some extent, shows the neural network’s capability. The more the hidden layers are, the higher feature it can extract and recognize from the input. People always use convolutional neural networks in computer vision, such as face recognition and image classification.

4.2. Long Short-Term Memory (LSTM) Networks

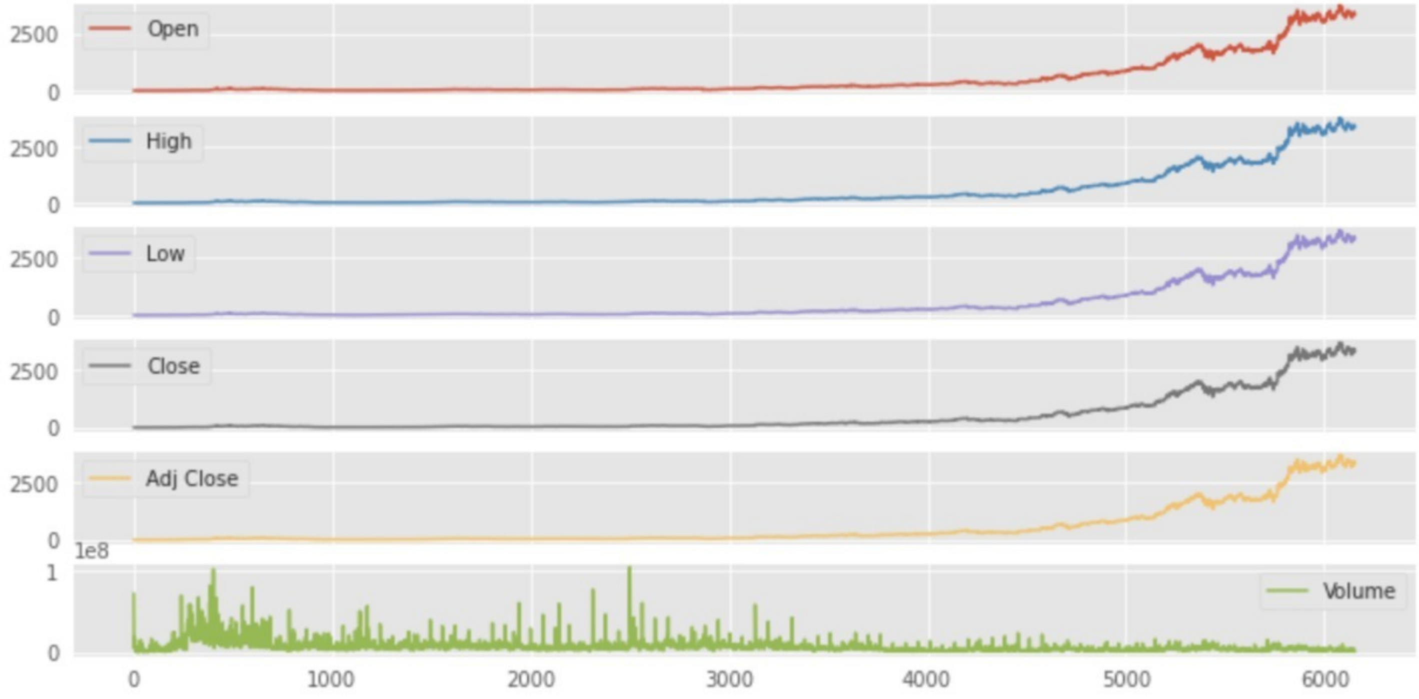
Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. This behavior is required in complex problem domains like machine translation, speech recognition, and more.



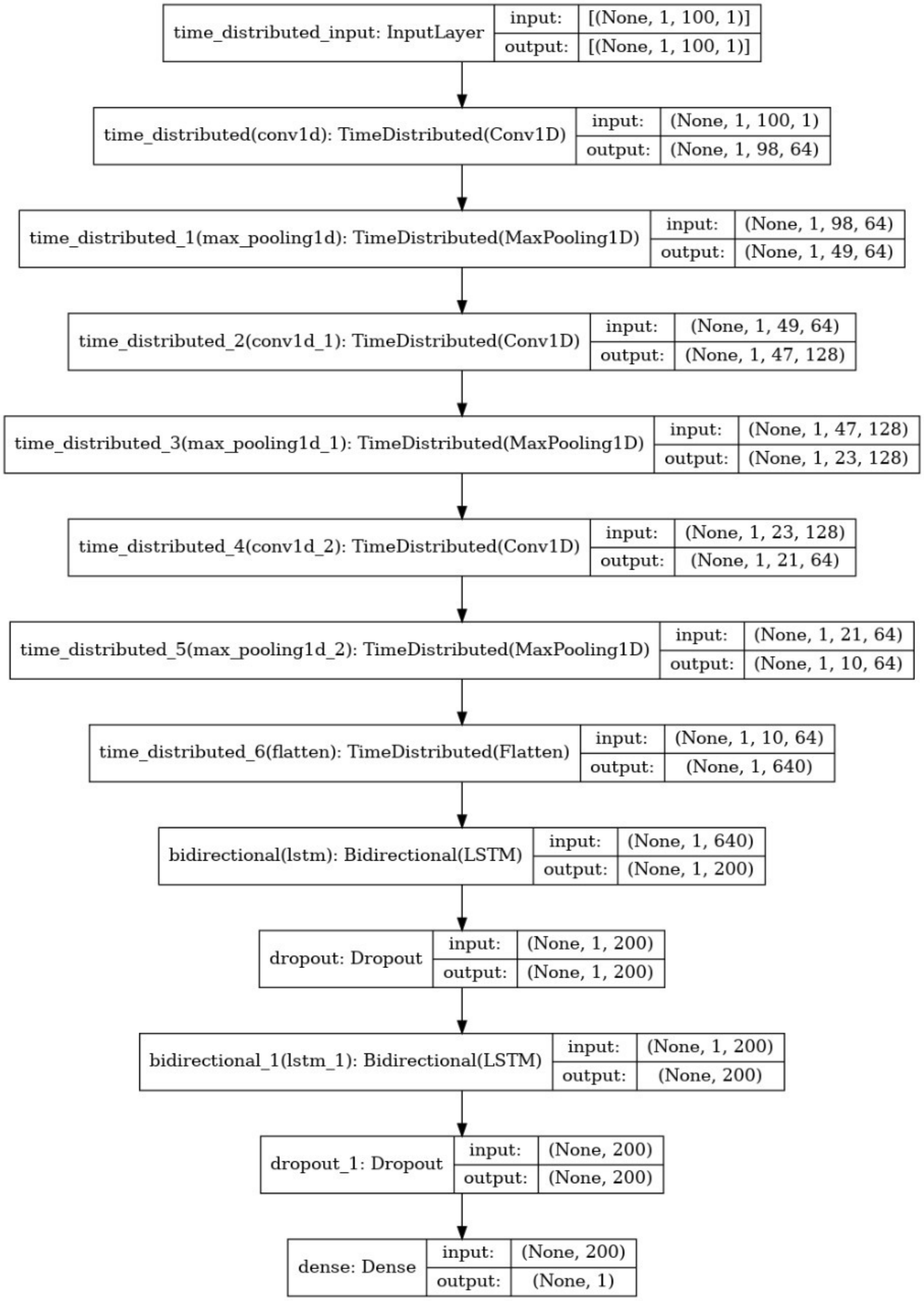
The picture above depicts four neural network layers in yellow boxes, point-wise operators in green circles, input in yellow circles, and cell state in blue circles. An LSTM module has a cell state and three gates that provide them with the power to selectively learn, unlearn, or retain information from each unit. The cell state in LSTM helps the information flow through the units without being altered by allowing only a few linear interactions. Each unit has an input, output and a forget gate which can add or remove the information to the cell state. The input gate controls the information flow to the current cell state using a point-wise multiplication operation of 'sigmoid' and 'tanh,' respectively. The forget gate decides which information from the previous cell state should be forgotten for which it uses a sigmoid function. Finally, the output gate decides which information should be passed on to the next hidden state.

4.3. Experiment

Visualization of the feature set:

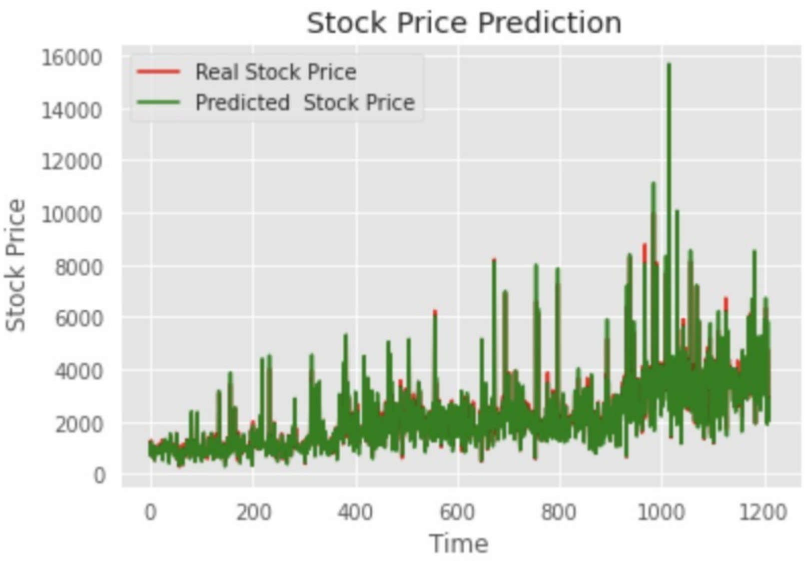


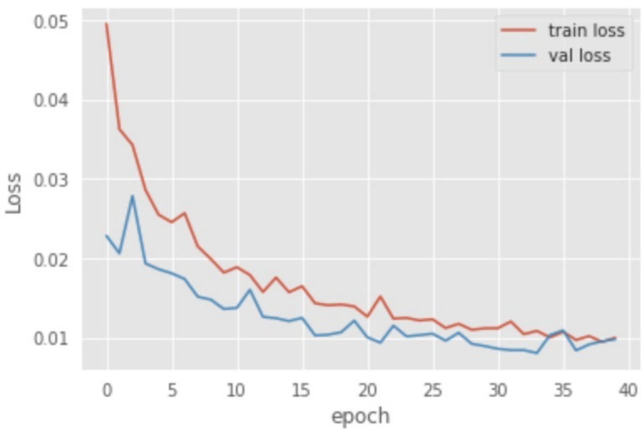
For CNN, the layers are created with sizes 64,128,64 with kernel size = 3. In every layer, the TimeDistributed function is added to track the features for every temporal slice of data regarding time. In between, MaxPooling layers are added. After that, it is passed to Bi-LSTM layers. The function we adopt in the model accepts some stock data and other properties such as the number of filters, the width of filters, and stride as its input. Then we use a Gaussian distribution to initiate the value of the filters and zero to initiate the biases. The outputs of our function are some matrices, and the number of the filters decides the specific number. CNN model will extract some features from them, and they also will be the input of the next pooling layer after the computation of the activation function. Using the conv1d function, we can build a CNN model to help us predict. To get the final classification values, which will tell us the stock will be up or down in the future, we will use the “teaching” number to train the model. Firstly, the vector of stock data will be input into the stock prediction model, and we will use the conv1d function to compute convolution. Because our stock prediction model may be linear and very poor to solve the nonlinear problem, it is necessary to introduce the activation function.

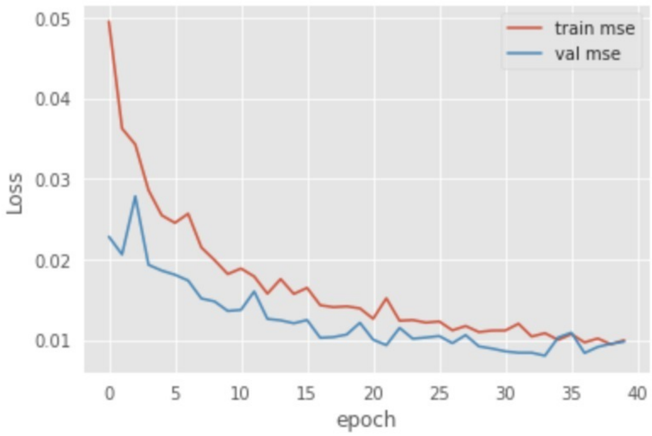

time_distributed_input: InputLayer
input: [(None, 1, 100, 1)]
output: [(None, 1, 100, 1)]
time_distributed(conv1d): TimeDistributed(Conv1D)
input: (None, 1, 100, 1)
output: (None, 1, 98, 64)
time_distributed_1(max_pooling1d): TimeDistributed(MaxPooling1D)
input: (None, 1, 98, 64)
output: (None, 1, 49, 64)
time_distributed_2(conv1d_1): TimeDistributed(Conv1D)
input: (None, 1, 49, 64)
output: (None, 1, 47, 128)
time_distributed_3(max_pooling1d_1): TimeDistributed(MaxPooling1D)
input: (None, 1, 47, 128)
output: (None, 1, 23, 128)
time_distributed_4(conv1d_2): TimeDistributed(Conv1D)
input: (None, 1, 23, 128)
output: (None, 1, 21, 64)
time_distributed_5(max_pooling1d_2): TimeDistributed(MaxPooling1D)
input: (None, 1, 21, 64)
output: (None, 1, 10, 64)
time_distributed_6(flatten): TimeDistributed(Flatten)
input: (None, 1, 10, 64)
output: (None, 1, 640)
bidirectional(lstm): Bidirectional(LSTM)
input: (None, 1, 640)
output: (None, 1, 200)
dropout: Dropout
input: (None, 1, 200)
output: (None, 1, 200)
bidirectional_1(lstm_1): Bidirectional(LSTM)
input: (None, 1, 200)
output: (None, 200)
dropout_1: Dropout
input: (None, 200)
output: (None, 200)
dense: Dense
input: (None, 200)
output: (None, 1)

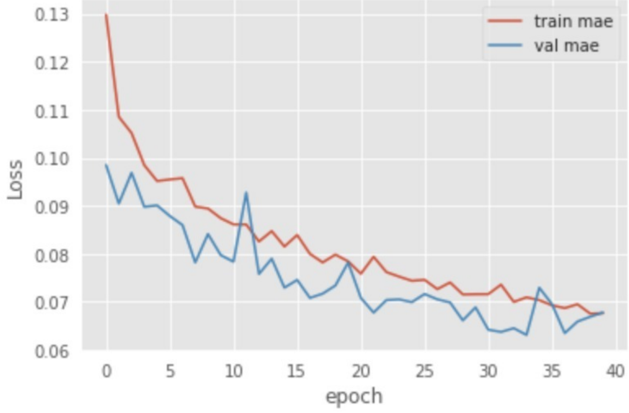
4. EVALUATION AND RESULTS:

All of the dataset's features (high, low, open, close volume, and adj close) were first used to train the LSTM and 1D-CNN models, then analyzed using the testing sets. Similarly, the trials were repeated with fewer features, after which the models were trained and evaluated. Mean square error (MSE) and mean absolute error (MAE) metrics were used to evaluate the proposed systems.









5. CONCLUSION AND FUTURE WORK

This paper introduced a CNN model to make the stock prediction and used a conv1d function to process the 1D data in the convolutional layer. We have also preprocessed stock data which will be input into the model to improve the model's result. Different stock data have evaluated the result and finally indicates that our CNN model is robust and can also be used to make the predictions even if the source data is 1D sequential. There are some exciting directions for further study:

(1) We can use the stock data with more features to make predictions, for the stock movement may not only be influenced by the features of open, close, high, and low prices. Moreover, we can compare which features are critical to the result.

(2) We can use more data sets to judge whether the result will be better, for the number of the data-sets is of great importance in deep learning.

(3) Using other convolutional neural network architectures can also be considered a good idea, including Google net and Alex net, for they have achieved good performance in ILSVRC. We can finetune them to take advantage of them to solve the problems in financial areas better.

REFERENCES:

1. https://www.kaggle.com/salmanfaroz/amazon-stock-price-1997-to-2020

2. Rasheed, Jawad, Akhtar Jamil, Alaa Ali Hameed, Muhammad Ilyas, Adem Özyavaş, and Naim Ajlouni. "Improving Stock Prediction Accuracy Using CNN and LSTM." In *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI)*, pp. 1-5. IEEE, 2020.

3. Chen, Sheng, and Hongxiang He. "Stock prediction using convolutional neural network." In *IOP Conference Series: materials science and engineering*, vol. 435, no. 1, p. 012026. IOP Publishing, 2018.

4. http://cs230.stanford.edu/projects_winter_2021/reports/70667451.pdf