Predicting Stock Price with a Long Short-term Memory Recurrent Neural Network

Introduction

Our Machine Learning application will attempt to understand how a stock price would develop in the future. Any stock's price is a dynamic variable that is formed in a "supply and demand" style manner, and it represents the market value for a company at a given time. This sort of application is desirable for obvious reasons. In this report, we will go through the steps needed to build a machine learning model. First, we form the problem outline. After the problem is outlined and assessed, we discuss methods and results. Lastly, we will summarize the outcome of our work. We are aiming for maximal effectiveness in prediction accuracy. An application domain for the model could be any organization that benefits from learning about stock price prediction, for example, an investment fund. This report does not guarantee any profit using the methods described below.

Problem Formulation

This project will be conducted as a supervised learning task. We will treat is as a recurrent neural network problem, where the goal is to predict the future stock price of INTC (Intel) based on historical data and relevant features. The predicted value will be continuous rather than a categorial one. Each historical datapoint represents four different values for a period, which is one stock exchange day in this case. The different data points represent values such as the opening and closing price, as well as the maximum and minimum values for the day in question. The Label for each datapoint is the stock price at a time instance. This is the value we even want to possibly be able to predict into the future. As the feature data, we can use any time instance price. The source for our time series dataset [1] is the United States public stock market, where data is constantly collected and reviewed by multiple parties. This kind of data is extremely reliable, meaning that it is also good training material for machine learning models.

Methods

We have an excel sheet with 1259 data points. These datapoints were taken one a day for five years, 2013-2018. In each datapoint you can see a date, highest and lowest stock price for the day, opening value, closing value, volume, and the name of the stock which in our case is INTC. INTC is the stock of tech company Intel. We chose stock prices and volume as our features because we are interested in the future stock value and volumes. We performed minimal feature engineering in this project, as the selected features are directly usable. However, we preprocessed the data by sorting it chronologically by date to ensure a consistent time series order. We chose to use all four daily prices because we thought it would yield the best results.

We chose LSTM, short for long short-term memory network, as our first ML method because it is one of the best approaches for regression analysis and time series forecasting [2]. As our second method we chose CNN (convolutional neural network). These methods have similarities as both are neural networks, and we thought it would be a clever idea to compare them with each other. But of course the two methods also have major differences and we wanted to see what kind of results these differences would yield.

LSTM is a recurrent neural network which is far more suitable for our project when compared to non-recurrent neural networks like CNN. We thought about four main points when choosing our methods. Firstly, stock price data is inherently sequential, where each data point's value depends on the previous data points. LSTMs are designed to model such sequences effectively. On the other hand, CNNs are designed for grid-like data such as images. LSTMs also have a mechanism to remember information over extended periods, which is crucial for capturing trends and patterns in historical stock prices. We also needed a method that could handle non-linear and complex patterns. Luckily, LSTMs can do this. Lastly both methods can automatically learn and extract relevant features from the historical stock price data, reducing the need for manual feature inputting.

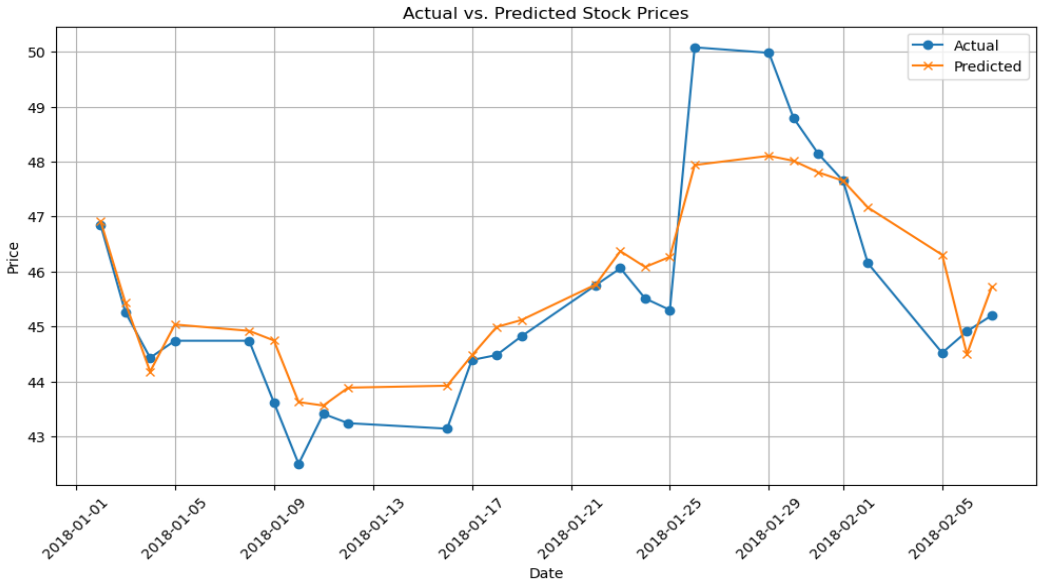
Our choice for the loss function for both methods is mean square error (MSE). MSE is great because it is sensitive to deviations. It penalizes large errors more heavily, which is important in financial forecasting. MSE is also a differentiable function, which makes it suitable for optimization algorithms like gradient descent, which is commonly used in training LSTMs. MSE is also a smart choice because it is so widely used, making it easy to compare our method to existing ones.

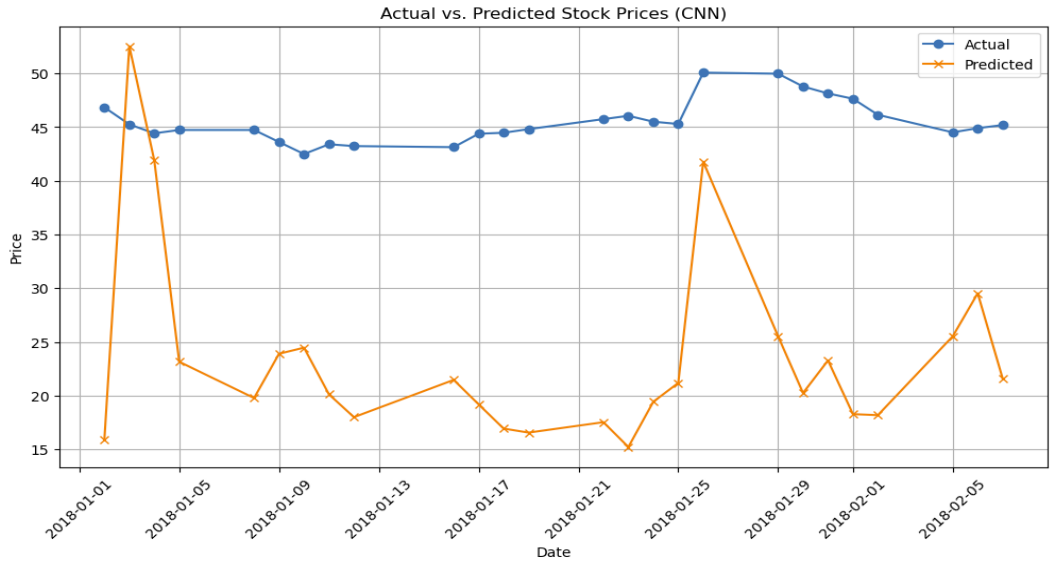
For this project we divided our dataset into just two parts. We split the available data 90% training and 10% testing. We do not include a validation set because in time series tasks like stock price prediction, chronological order matters significantly. If you choose not to use a validation set, you can train your model on all available past data, which may help capture long-term dependencies. Because of this, the size of the training set would be the first 1133 data points, and test set the last 126.

Results

Training loss for both methods are shown in each epoch trained. For the LSTM the training loss got smaller every epoch, but for the CNN the training loss had a wave like pattern getting smaller and then larger circling around 400. The training loss for the whole training for LSTM was 0.75 and 560 for CNN, which is an enormous difference. This indicates that the CNN model had a much higher error rate when making predictions on its training data and that LSTM learns much better given the same data. In general, a lower training loss can be an indicator of the model's ability to fit the training data well. Since we had no validation set, we had no validation errors.

Test errors can be seen with plots that we graph. We test the data by comparing the machine's predicted prices with the test data set consisting of the last 126 days of our data. The closer the predicted graph is to the actual the better. Here are the test graphs of both LSTM and CNN





As we can see the LSTMs predicted prices follow the actual prices significantly better than the CNNs. LSTMs final test loss was 0.753 and CNNs was 560. This is perhaps happening because of the training loss or just because LSTM is a better alternative for this project. Both training and test losses are smaller for LSTM so choosing the final method is easy. Our final choice is LSTM.

Conclusion

The report investigated training two similar but fundamentally different ML methods. First, a kind of recurrent neural network LSTM model was trained. The resulting error from the test evaluation was lower than expected as we also expected the training to be more challenging. The results shown in Appendix A confirm overall positive results in test loss, after training was completed. After training the LSTM, another model of CNN, was trained. The training occurred in a similar manner. The CNN model was also evaluated through test loss(11) after training. The results were a lot worse than with the first LSTM model, which was expected, as an RNN model was recommended to be used in such cases [2]. Furthermore, even

though the LSTM model was much more accurate, the CNN model was superior in terms of training time, taking a fraction of the time to train the model. The results from both have room for improvement, as one can always train a model with a larger dataset, with more dimensions, having more labels to learn on.

Frankly, stock prices are difficult or impossible to predict, and to this day, no complete prediction model has been built. Prediction of stocks includes such a huge number of labels with different forms of data, that collecting them in real-time, and constantly training a model with them, is at least a very time-consuming and challenging task. The future is also thought to be inherently unpredictable.

References

[1] Nugent, C (cited 10th Sept. 2023). *"S&P 500 Stock Data"*: Kaggle. https://www.kaggle.com/datasets/camnugent/sandp500?resource=download

[2] Kharwal, A (cited 10th Sept. 2023). *"Netflix Stock Price Prediction with Machine Learning"*: thecleverprogrammer.com. https://thecleverprogrammer.com/2022/02/08/netflix-stock-price-prediction-with-machine-learning/