Stock Market Price Prediction using LSTM

Kerem Haktan Kurt

October 2024

1 Introduction

The project aims to predict a Stock's future price based on its previous data. Using Yahoo Finance to download its past 10 years of data. By analyzing data, selecting a model and evaluating the model.

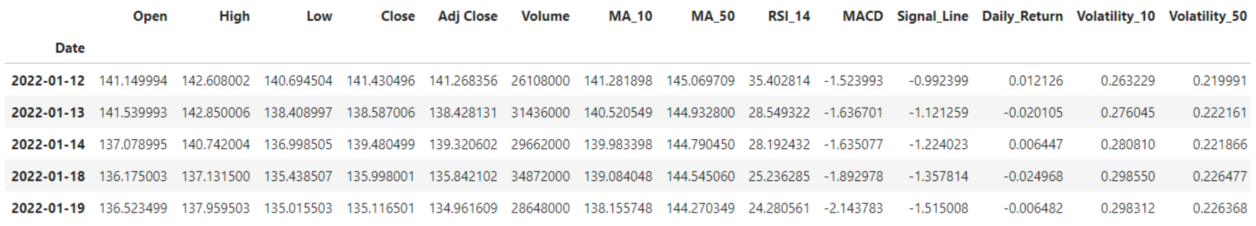
2 Data Description

At first our data had 2516 rows and 6 columns, from this starting point I added the features : Moving Average for both 10 and 50 days, RSI for 14 days, MACD, Signal Line, and lastly Volatility for 10 and 50 days.

But then I needed to trim the rows with None columns if they had any and erased the first 50 rows. Resulting in the final table for my train test split: 2466 x 14

Then split my dataset for 80% for training and 20% for testing. This means that the train data will have the data from 2014 to 2022 and the test data will have from there on. By doing this and not shuffling, I keep the data sequentially because in the financial market, the price is connected to what it was before, meaning shuffling would make a mass and we don't want that.

Figure 1: The Dataset



	Open	High	Low	Close	Adj Close	Volume	MA_10	MA_50	RSI_14	MACD	Signal_Line	Daily_Return	Volatility_10	Volatility_50
Date														
2022-01-12	141.149994	142.608002	140.694504	141.430496	141.268356	26108000	141.281898	145.069709	35.402814	-1.523993	-0.992399	0.012126	0.263229	0.219991
2022-01-13	141.539993	142.850006	138.408997	138.587006	138.428131	31436000	140.520549	144.932800	28.549322	-1.636701	-1.121259	-0.020105	0.276045	0.222161
2022-01-14	137.078995	140.742004	136.998505	139.480499	139.320602	29662000	139.983398	144.790450	28.192432	-1.635077	-1.224023	0.006447	0.280810	0.221866
2022-01-18	136.175003	137.131500	135.438507	135.998001	135.842102	34872000	139.084048	144.545060	25.236285	-1.892978	-1.357814	-0.024968	0.298550	0.226477
2022-01-19	136.523499	137.959503	135.015503	135.116501	134.961609	28648000	138.155748	144.270349	24.280561	-2.143783	-1.515008	-0.006482	0.298312	0.226368

3 Data Preprocessing

My preprocess follows the order of:

-Downloading the Data

-Creating Features(Indicators)

-Handling missing values

-Split the data

Then initialized my scaler to scale both the training and test dataset. We need scaling because not all features are at the same scale and this may result in the model giving more importance to the ones that have a larger magnitude. If we were to compare RSI, MACD, and Volatility the model may give more importance to the RSI because sometimes the magnitude of RSI may reach 100 times the Volatility, and this is not ideal for our data to train the model. I only train the futures that I will use for training

4 The Model

- **Long Short-Term Memory (LSTM)**: LSTM is designed specifically to predict sequential data, and stock market prices are always sequential one by another. LSTM's can understand time dependency better than other models and handle many indicators at once. Random Forest and Linear Regression models don't have time dependency so they won't be suited for this job

  Inputs: Close, MA-10, MA-50, RSI-14, MACD, Signal-Line, Volatility-10, Volatility-50

  Otput: The Close price

  50 neurons in each hidden state, and 2 LSTM Layers.

  Meaning that this is a pretty small and efficient model to train. Even though the model is small the performance is nice.

5 Training

The Visualization of the Training Process: Wandb Run

I have experimented with many sequence lengths for prediction and found out that the best performing number was 50. And continued the process with it. Tried many learning rates and various scheduler to find out the most effective one and found that learning rate of 5e-5 and cosine scheduler seemed to be working the best.

The training is done for 100 epoch with early stopping meaning that the model does not overfit on the training data. And can stop early if such thing starts to happen.

- At the beginning of each epoch, the loss is set to zero and lists for predictions and actual target values are initialized.
- For each bach the gradients are cleared and the sequence is feed to the model to get the prediction, loss is computed and it got compared with the target.
- Gradients are calculated with loss.backward() and stepped up the optimizer to update the model weights.
- Losses and error values (MSE, RMSE, MAE) is calculated
- The training process is logged onto the wandb run
- Checking for early stopping at the end of each epoch

6 Evaluation

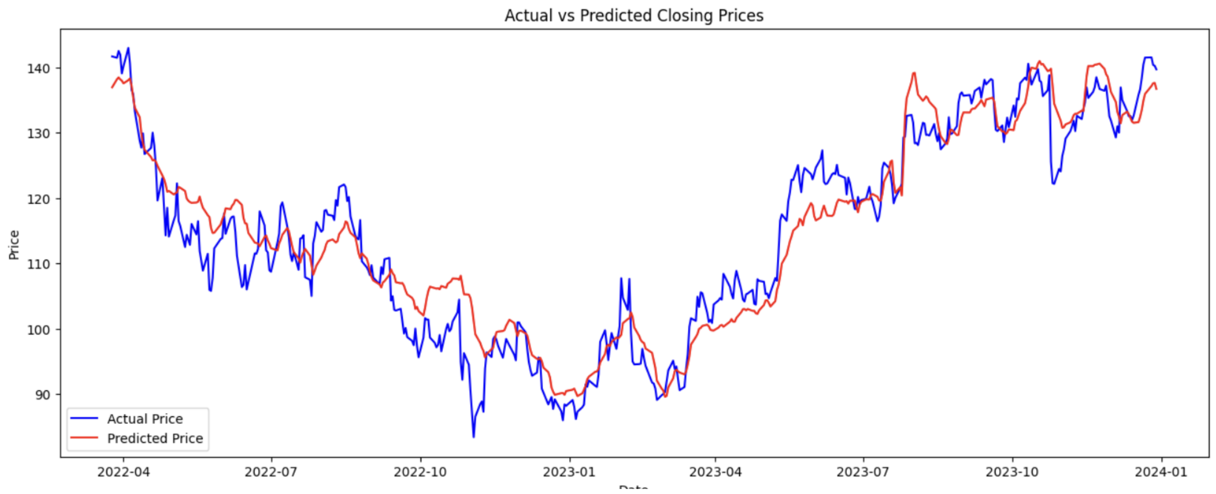
Evaluation is also done after each epoch to check if the model betters itself or is at a plateu. Evaluation is done with data from 2022-04 to 2024-01, and the model got the error values after the training to:
Evaluation criteria are:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad \text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad \text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

7 Results

Mean Squared Error (MSE): 22.54692996507974 Root Mean Squared Error (RMSE): 4.748360766104419
Mean Absolute Error (MAE): 3.7476038164566634

Figure 2: The Prediction vs the Actual



8 Conclusion

This project showcases a successful implementation of an LSTM model for future stock price prediction model. Managing to work with a dataset of 2500 rows with over 10 columns. Researched many areas throughout the development process of this model to accurately predict what to use instead of spending resources testing trivial ones. Keeping the valuable resources for what may cause conflicts and selecting the best one.

9 References

- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation.
- McKinney, W. (2010). Data structures for statistical computing in Python. In *Proceedings of the 9th Python in Science Conference* (Vol. 445, pp. 51-56). `https://pandas.pydata.org`.
- Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32, 8024-8035. `https://pytorch.org`.
- EDA guide
- Transformers vs LSTM for stock price prediction
- Many more articles and artificial intelligence (I have written only the big ones that looked a lot).