STOCK TRADING TOOL

ABSTRACT

In this project, we have attempted to build a model where a user can visualize the trend of each stock and make an informed decision before trading. We are using machine learning approach to forecast stock prices. The objective of this project is to design a user interactive model where a user has the option to trade or explore and can select any stock within the database to visualize the prediction of that the stock's prices to make accurate investment decisions based on the data from the last 4 years.

Here, we have implemented the Long Short-Term Memory (LSTM) model of RNN (Recurrent Neural Network) to predict the stock closing prices on the 1-minute dataset of 5 Indian companies starting from 2017–01–01. For analyzing the stock price trend, we are using a Time series plot, and to check the model accuracy, RMSE (Root Mean Squared Error) is used.

INTRODUCTION

The objective of this project is to understand the stock price trend over a period and the factors that are impacting the stock performance and how we can leverage it. We have used this information to build a model which can make a prediction of the direction and magnitude of the changes in the stock price in near future and will suggest user which stock to buy or hold.

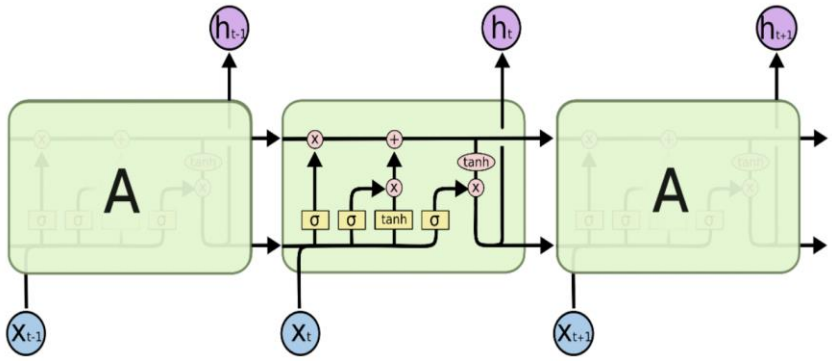
For our project, we have obtained the stock data from Kaggle over a period of four years (2017-2021), for 5 companies to assess their stock prices.

MODEL

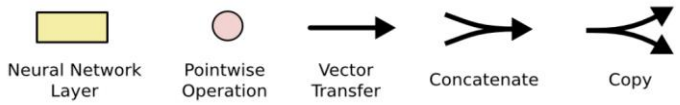
LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is that LSTM can store past important information and forget the information that is not.

LSTM has three gates:

- The input gate: The input gate adds information to the cell state,
- The forget gate: It removes the information that is no longer required by the model,
- The output gate: Output Gate at LSTM selects the information to be shown as output.

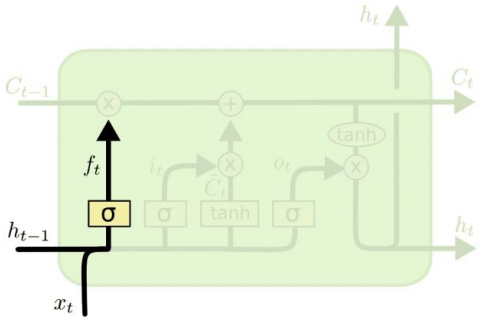


LSTM workflow



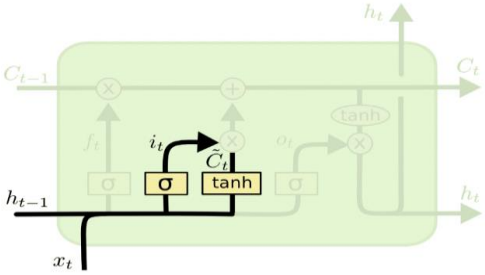
Notations

The first step in our LSTM is to decide what information we're going to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer." It looks at h_{t-1} and x_t, and outputs a number between 0 and 1 for each number in the cell state C_{t-1}. 1 represents "completely keep this" while a 0 represents "completely get rid of this."



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

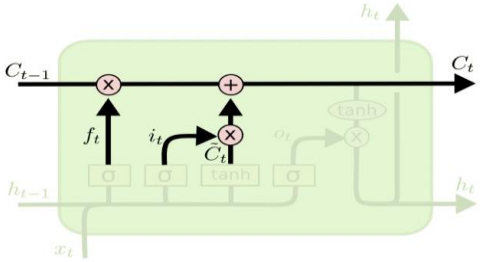
The next step is to decide what new information we're going to store in the cell state. This has two parts. First, a sigmoid layer called the "input gate layer" decides which values we'll update. Next, a tanh layer creates a vector of new candidate values, $\tilde{C}_t$, that could be added to the state. In the next step, we'll combine these two to create an update to the state.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

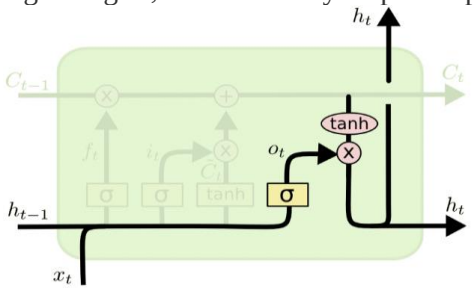
It's now time to update the old cell state, C_{t-1}, into the new cell state C_t. The previous steps already decided what to do, we just need to do it.

We multiply the old state by f_t, forgetting the things we decided to forget earlier. Then we add $i_t * \tilde{C}_t$. This is the new candidate values, scaled by how much we decided to update each state value.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Finally, we need to decide what we're going to output. This output will be based on our cell state but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to output. Then, we put the cell state through tanh (to push the values to be between −1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

DATA

We have taken four-years data for 5 companies (Bajaj Auto, Fortis, HDFC Bank, Pfizer, TCS) and loaded into the database from the CSV files after cleaning and normalizing. Each file consists of more than 400k rows with variables such as Year, Date, Time, Open_Price, Close_Price, High_Price, Low_Price, Volume, Timestamp. We have trained 80% of the data using LSTM Model for Stock Market Prediction.
We have used multiple interdependent functions to build the interactive user interface, which is helping user to choose the various options for stock trading followed by the machine learning LSTM model which will give the prediction and comprehensive report of the selected stock.

DATA VISUALIZATION

To plot price history visualization of the selected stock over the years, we have used in-built python visualization libraries such as plotly.

ANALYSIS

We have trained 80% of the data using LSTM Model for Stock Market Prediction. Then, validating our results over remaining 20% of test data using visualization tools such as plotly. To check the model accuracy, we are using Root mean squared error (RMSE).

RESULTS

Model Performance:
RMSE for all 5 stocks:
BAJAJ AUTO-8.79232208479227
PFIZER- 21.01025996381319
TCS- 17.010158573814145
FORTIS- 0.35774330347772976
HDFCBANK- 3.270899644095874

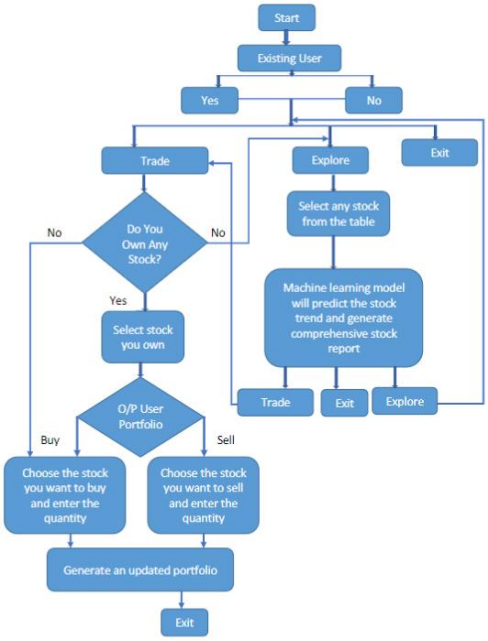
Stock Trend Forecast

Below graphical image shows stock price predicted trend on test data for all the five stocks

TOOL FLOWCHART

We have built a Trading tool that an user can use to Explore stocks and Trade i.e. Buy and Sell stocks. Users can build their portfolios for future Trading purposes. We have tested most of the possible test-cases within the Tool and we have plans to expand the tool for wider use-cases.

Below is the Flowchart:



CONCLUSION

Using the LSTM Machine Learning model, we can predict the closing stock price, and using this we can make informed decision whether to Buy, Hold or Sell the stock.

FUTURE RESEARCH DIRECTIONS

- Currently, our model is learning from train datasets and predicting over test data. As a future enhancement, we can build a feature that will predict an approximate forecast for the future stock price.
- We know that the scale of a company impacts the volatility of the day-to-day stock price. In this model, we have not taken that into account. This can also be a future enhancement for the model
- The Database currently fully loads data for all CSVs. As an enhancement, we can implement Delta loading, where only new records will be loaded every time the Database query is run. This can be implemented using the Timestamp column.
- We can add a feature where the model will suggest to users which are the best-suited stocks to buy/sell/hold, and most active stocks, and which stock options to ignore for that time.
- Improving the precision of the model by incorporating other more informative features.

REFERENCE

- https://www.kaggle.com/datasets/hk7797/stock-market-india
- http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://www.moneycontrol.com/india/stockpricequote/computers-software/tataconsultancyservices/TCS