

**MOSAIC ‘24:**

**STOCK CLOSE PRICE PREDICTION**

**TASK DESCRIPTION:**

Our task is to predict the stock close price for the upcoming 96 days for all the 6 companies from the given data about which comprises of 8 columns (excluding index column) namely:

Open: The starting price of a stock on a particular day.

High: The highest price the stock reaches during that day.

Low: The lowest price the stock drops to during that day.

Volume: The total number of shares traded within that day, indicating market activity.

Close: The final price of the stock at the end of the trading day.

Adjusted Close: This considers factors like dividends and stock splits to give a more accurate picture of the stock's value.

Company: The name of the company

**ANALYSIS OF THE DATASET:**

Exploratory Data Analysis (EDA) was conducted to gain insights into the dataset's structure, distribution, and relationships among variables.

* Checking for multicollinearity: Correlations between Open, High, Low, Close, and Volume stock prices for six companies were analysed through heatmaps, revealing strong interdependencies. To mitigate multicollinearity, Open, High, and Low price columns will be dropped, streamlining the dataset for further analysis
* Closing Price Analysis :

The analysis begins with a check for null values in the dataset using `df\_closing.info()`. Subsequently, a grid of subplots is

created to visualise the closing prices of six companies, each plotted against day number. This visualisation offers insights into the closing price behaviour of each company over time, aiding in the examination of market trends and performance.

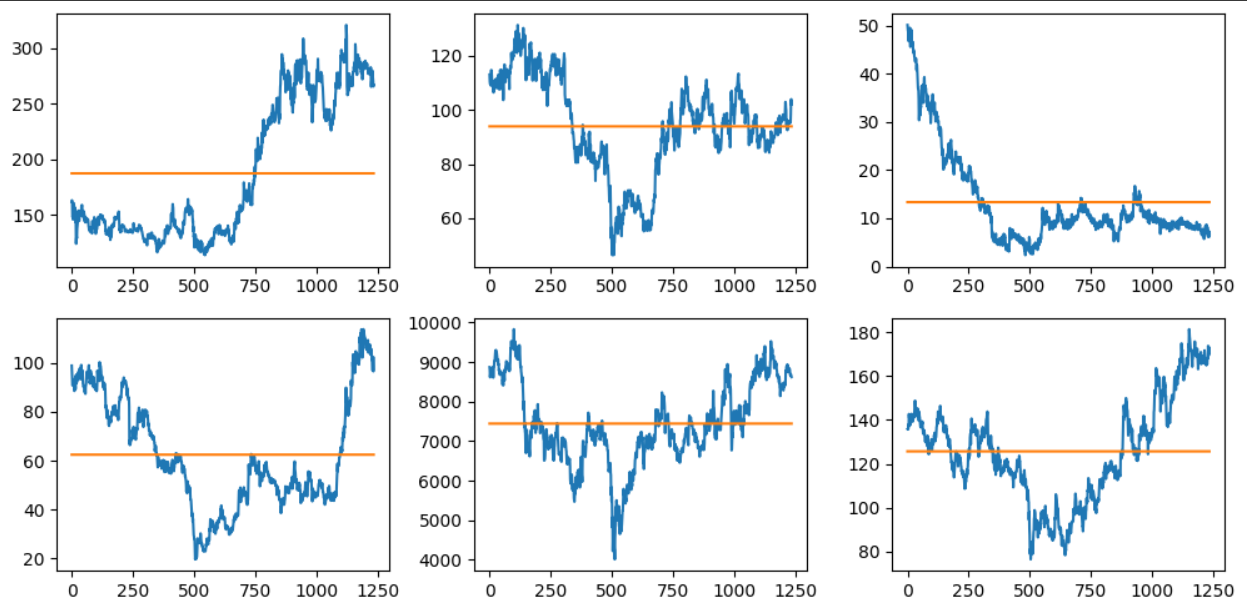


Careful training is essential for Company\_4 due to its wide

range of close prices, minimising the risk of high RMSE.

* Examining Variation with help of mean :

The closing prices are plotted against time, with a horizontal line representing the mean closing price across all companies. This visualisation aids in comparing individual company trends to the overall market behaviour.



* Outlier Detection and its effect :

The boxplots illustrate the central tendency, spread, and variability of closing prices over different years for each company, aiding in trend analysis and outlier detection. Although some outliers are present in certain years, their impact on the overall dataset appears negligible, ensuring robustness in our analysis.

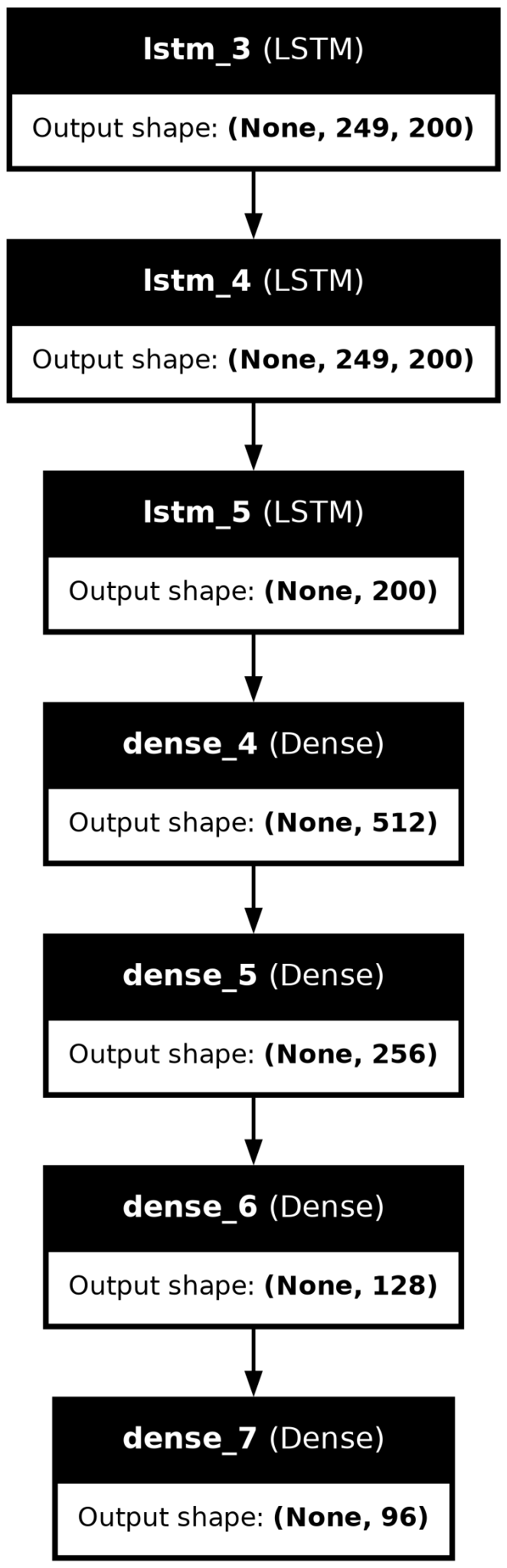
**OUR APPROACH:**

On the above analysis of the dataset we first pre-process the data, we decided to dump all the columns except close\_price of each company and then separated the close\_price based on the company due to the fact that it is a time series forecasting and hence it is predominantly dependent on it’s past values rather than the other columns, and then even check for missing values and all.

We then split into X and y from where X is a 2d array containing n\_lookback elements up to the whole close column , thus dimensions of X: [n\_lookback, n-n\_lookback+1] &

dimensions of y: [n\_forecast, n-n\_lookback+1] .

Then we implement neural models as shown below:



Here we used 3 Layers of LSTM followed by

3 layers of Dense layer decreasing output shape till it reached the final y.shape , we have used leaky ReLU as our activation unit in it .

And for better and faster descent used Adam as our optimizer .

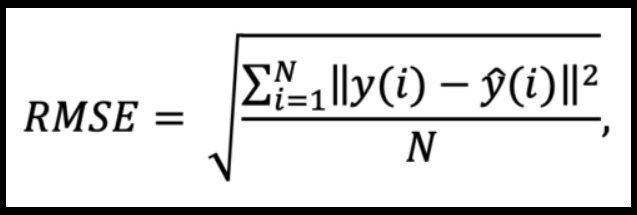
We reached this final model after several iterations of the training models where we analysed the plot for each to get to know the nature of the plot and also we had split the data into train and validation in ration 80:20 where we seek the least validation for fewer epochs

And hence after tuning the hyperparameters that work for our benefit , it became our final model

(refer to colab link provided in last page)

**PREDICTIONS:**

Prediction of the output 96 days data is judged under the RMSE (root mean squared error) between the actual close prices and predicted closing prices.



Public score of 90.6748

Private score of 255.2538

**SHORTCOMINGS OF OUR APPROACH:**

While our approach of using LSTM for stock price prediction has its strengths, it also has several shortcomings that should be acknowledged:

1.Limited Feature Set: Focusing solely on past stock prices as input features neglects potentially valuable information from other relevant factors such as trading volume, market sentiment, news events, and macroeconomic indicators. Ignoring these features may result in a less comprehensive model that fails to capture all relevant market dynamics.

2.Data Dependence: LSTM models are highly dependent on the quality and quantity of historical data. If the available dataset is limited or lacks diversity, the model may struggle to generalise well to unseen data or future market conditions.

3.Overfitting: LSTM models, especially when trained on limited data, are prone to overfitting, wherein the model learns to memorise the training data rather than capturing underlying patterns. Overfitting can lead to poor performance when applied to new data.

4.Complexity and Computational Cost: LSTM models are computationally intensive and may require significant computational resources, especially when dealing with large datasets or complex architectures. This can result in longer training times and higher computational costs.

5.Limited Interpretability: Deep learning models like LSTM are often considered "black-box" models, meaning that it can be challenging to interpret how the model arrives at its predictions. Lack of interpretability may hinder understanding of the factors driving the model's decisions, which is crucial for stakeholders and decision-makers.

6.Assumption of Stationarity: LSTM models assume that the underlying time series data is stationary, meaning that statistical properties such as mean and variance remain constant over time. However, financial time series data often exhibits non-stationary behaviour, such as trends, seasonality, and volatility clustering, which may violate this assumption and impact model performance.

7.Limited Forecast Horizon: LSTM models are generally better suited for short to medium-term forecasting horizons. Attempting to forecast too far into the future may lead to increasingly inaccurate predictions due to the accumulation of errors over time.

8.Sensitivity to Hyperparameters: LSTM models require careful tuning of hyperparameters such as the number of layers, number of units, learning rate, and dropout rate. Suboptimal hyperparameter choices can significantly affect model performance and may require extensive experimentation to find the best configuration.

**FUTURE SCOPE AND FURTHER IMPROVEMENT:**

Using LSTM (Long Short-Term Memory) for predicting stock prices is a promising approach due to its ability to capture long-term dependencies in sequential data. However, there's always room for improvement and further exploration. Here are some suggestions for the future scope and enhancements:

1.**Feature Engineering**: Expand the feature set used for prediction beyond just historical stock prices. Incorporate additional data such as trading volume, market sentiment, macroeconomic indicators, news sentiment, or technical indicators like Moving Averages, Relative Strength Index (RSI), and Bollinger Bands. These additional features could provide more context and potentially improve the model's performance.

2.**Hyperparameter Tuning**: Experiment with different hyperparameters of the LSTM model such as the number of LSTM layers, the number of neurons in each layer, learning rate, batch size, and dropout rate. Hyperparameter optimization techniques like grid search or random search could help identify the optimal set of hyperparameters for your specific dataset.

3.**Model Architecture**: Explore alternative neural network architectures beyond LSTM, such as GRU (Gated Recurrent Unit), CNN-LSTM (Convolutional Neural Network followed by LSTM), or Transformer-based models like GPT (Generative Pre-trained Transformer). Each architecture has its strengths and weaknesses, and experimenting with different architectures could lead to performance improvements.

4.**Ensemble Methods**: Combine predictions from multiple models to improve overall performance. Techniques like model averaging or stacking can help reduce model variance and improve generalisation.

5.**Data Preprocessing**: Pay close attention to data preprocessing steps. Ensure that data is properly scaled, normalised, and cleaned before feeding it into the model. Consider techniques like feature scaling, differencing, or normalisation to make the data more suitable for training.

6.**Regularization**: Implement regularisation techniques such as L1 or L2 regularisation, dropout, or early stopping to prevent overfitting and improve the model's generalisation ability.

7.**Sequential Patterns**: Investigate the presence of any specific sequential patterns or dependencies in the data and design the model architecture accordingly. This could involve experimenting with attention mechanisms or memory-augmented neural networks.

8.**Model Interpretability**: Enhance the interpretability of the model by incorporating techniques such as attention mechanisms or layer-wise relevance propagation (LRP) to understand which features or data points are most influential in making predictions.

9.**Data Augmentation**: Augment the training data using techniques like time-series data augmentation or synthetic data generation to increase the diversity of the training dataset and improve the model's robustness.

10.**Deployment and Monitoring**: Develop a robust deployment pipeline for the model and continuously monitor its performance in a real-world setting. Implement mechanisms for model retraining and updating to adapt to changing market conditions.

By exploring these avenues for improvement, we can enhance the predictive capabilities of your model and potentially achieve better accuracy in forecasting stock prices.

**Colab link:**

<https://colab.research.google.com/drive/1rlmam0DbM06pl3xWQZwRq7z2IlfakDLJ?usp=sharing>

**BABY DRIVER  
Aryansh Kumar**

**Mitul Agarwal**

**Aditya Raj**