**Stock Price Prediction**

Alright, so here’s what we did in this notebook. We wanted to predict stock prices using an LSTM model, and we followed a step-by-step approach to get there.

**1. Importing Required Libraries**

First, we imported all the necessary Python libraries like Numpy, Pandas, Matplotlib, Seaborn, and Tensor Flow. We also used yahoo finance to fetch stock data. These libraries help us with data handling, visualization, and building the LSTM model.

**2. Loading the Data**

We loaded the stock market data from an Excel file and checked what it looks like using .head() and .describe(). Also, we checked for missing values and found that there were no missing values in our dataset.

**3. Plotting the Stock Prices**

To understand the trend of stock prices over time, we plotted the closing price using Matplotlib. This gave us a visual understanding of how stock prices have changed over time.

**4. Feature Selection & Data Preprocessing**

We selected the most relevant ones: Open, High, Low, Close, and Volume. Then, we normalized the data using MinMaxScaler. This helps in keeping all values between 0 and 1, making it easier for the model to learn patterns and improve performance.

**5. Creating Training Data for LSTM**

We prepared the input sequences for LSTM. We took past 60 days of stock prices to predict the next day’s closing price. Since LSTM expects input in a specific shape, we reshaped the data accordingly.

**6. Building the LSTM Model**

Now, we built our LSTM neural network. We used:

* Two LSTM layers to capture patterns.
* Dropout layers (20%) to prevent over fitting.
* Dense layers to make the final prediction.

Then, we compiled the model using Adam optimizer and Mean Squared Error loss function.

**7. Training the Model**

We trained the model using 10 epochs with a batch size of 20. The validation set was set to 10% of the training data.

**8. Predicting the Next Day’s Stock Price**

To make a prediction, we took the last 60 days of stock data, reshaped it, and passed it into the trained model. Since the prediction is in the scaled format, we converted it back to actual stock prices using inverse transformation.

**9. Evaluating the Model**

We plotted the training loss vs. validation loss to see how well our model is performing. Finally, we compared actual vs. predicted stock prices to check how well our model captures market patterns.

**Conclusion**

So, in this project, we trained an LSTM model to predict stock prices based on historical data. We used past 60 days of data to predict the next day’s closing price. The model captures trends fairly well, but improvements can be made by tuning parameters or adding more features like technical indicators. Overall, this was a great way to see how deep learning can be applied to financial markets!