# Stock Price Prediction using LSTM with Hyperparameter Tuning

## ****1. Introduction****

This project focuses on predicting stock prices using a Long Short-Term Memory (LSTM) model. LSTMs are a type of recurrent neural network (RNN) that are well-suited for time-series forecasting. The model is built and optimized using **KerasTuner**, which helps in selecting the best hyperparameters for improved performance.

## ****2. Data Preprocessing****

### ****2.1 Scaling the Data****

Stock price data is first normalized using **MinMaxScaler** with a feature range of (0,1). This ensures that all values are within the same scale, preventing large numbers from dominating the training process.

### ****2.2 Creating Sequences****

The model uses past **60 days** of stock prices to predict the next day’s closing price. The dataset is transformed into a format where each input sample contains 60 consecutive days of prices, and the output is the price for the next day.

### ****2.3 Splitting the Data****

The dataset is divided into **80% training data** and **20% test data** to evaluate the model's performance.

## ****3. Model Architecture****

### ****3.1 LSTM Layers****

* Two **LSTM layers** are used, each with a tunable number of units (ranging from 50 to 128).
* The first LSTM layer has return\_sequences=True to pass the output to the next LSTM layer.
* The second LSTM layer has return\_sequences=False as it outputs the final representation.

### ****3.2 Dropout Regularization****

* A **Dropout layer** (tunable between 0.2, 0.3, and 0.4) is added after each LSTM layer to prevent overfitting.

### ****3.3 Dense Layers****

* One **Dense layer** is added with tunable units (between 25 and 100).
* The output layer consists of a single neuron that predicts the stock price.

### ****3.4 Optimizer and Loss Function****

* The model uses the **Adam optimizer**, with a tunable **learning rate** (0.001, 0.0005, 0.0001) to achieve efficient gradient descent.
* The **Mean Squared Error (MSE)** loss function is used, as it is suitable for regression problems.

## ****4. Hyperparameter Tuning using KerasTuner****

Hyperparameter tuning is performed using **Hyperband**, which efficiently searches for the best combination of parameters:

* Number of LSTM units
* Dropout rate
* Number of Dense units
* Learning rate

The best hyperparameters are selected based on the lowest **validation loss**.

## ****5. Model Training and Evaluation****

* The model is trained for **50 epochs** using the best hyperparameters obtained from tuning.
* It is evaluated on the test set using the **MSE loss**.
* Predictions are made on the test set and **inverse transformed** to the original scale.

## ****6. Results and Visualization****

* A plot is generated comparing **actual stock prices vs. predicted prices**.
* The visualization helps in understanding the accuracy of the model in capturing stock price trends.

## ****7. Conclusion****

* LSTM models perform well for stock price forecasting by capturing sequential dependencies.
* Hyperparameter tuning significantly improves model performance.