# Introduction

In this project, we used a long short-term memory (LSTM) neural network to analyze time series data and generate a forecast for future values. The dataset used in this project is data.csv, which contains the daily closing prices of a stock.

# Data Preparation

The first step was to load and prepare the data. We used the pandas library to load the data from the CSV file, and then converted it to a numpy array. We then split the data into training and testing sets, with a split ratio of 0.8. The training data was used to train the LSTM model, while the testing data was used to evaluate the performance of the model.

We also normalized the data using a MinMaxScaler from the scikit-learn library. Normalization is an important step when training neural networks, as it helps to ensure that all the input data is within a similar range, which can help the model to learn more effectively.

# Model Building

Next, we built the LSTM model using the Keras library. The model had one LSTM layer with 50 neurons, followed by a dense output layer with one neuron. We also added a dropout layer to help prevent overfitting.

We then compiled the model using the Adam optimizer and the mean squared error loss function. The mean squared error loss function is commonly used in regression problems, and is a measure of how well the model is able to predict the output values.

# Model Training

The next step was to train the model using the training data. We trained the model for 100 epochs, and used a batch size of 32. During training, we monitored the mean squared error on the validation data to ensure that the model was not overfitting.

After training, we evaluated the performance of the model on the testing data. The mean squared error on the testing data was relatively low, indicating that the model was able to generalize well to new data.

# Forecasting

Finally, we used the trained model to generate a forecast for future values of the time series. We first created a list of the last window\_size data points from the training data, where window\_size is the number of time steps that the model takes as input. We then used the model to predict the next n values, where n is the number of time steps in the forecast.

To generate the forecast, we used a loop to iteratively predict the next value in the time series. For each iteration, we used the last window\_size data points as input to the model, and used the predicted value as input to the next iteration. We repeated this process for n iterations, and stored the predicted values in a list.

We then scaled the predicted values back to the original range using the inverse transform of the MinMaxScaler, and plotted the forecasted values alongside the actual values from the testing data.

# Conclusion

In conclusion, we were able to successfully build, train, and test an LSTM model for time series analysis, and generate a forecast for future values. The model was able to learn patterns in the training data and generalize well to new data, as indicated by the low mean squared error on the testing data. The forecasted values were also visually similar to the actual values, indicating that the model was able to capture the underlying patterns in the time series.