

Apple Stock Predictions Using LSTM - Documentation

Importing necessary libraries: pandas, NumPy, Matplotlib, Scikit-learn, and TensorFlow.

Approach-1:

Data-Preprocessing:

- Loading the dataset into the dataframe.
- Dataset shape: (9823, 7) — 9823 rows and 7 columns.
- Getting basic info of the dataset.
- The "Date" column is of object type, so it is converted to datetime format using pandas.
- The "Date" column is set as the **index** for time-series prediction.
- Checking Null values (1 Null Values).
- Used forward fill to handle missing values, as dropping them is not ideal for sequential stock price data.

Scaling the data:

- Used Min-Max scaler, as neural networks works best for scaled values.
- Setting feature range between **0** to **1**. And selected the target variable as 'Adj Close'

Adj Close	
Date	
1980-12-12	0.000943
1980-12-15	0.000863
1980-12-16	0.000757
1980-12-17	0.000790
1980-12-18	0.000830

Creating Time-sequences:

- Defined a function to create sequences.
- Applied that function into the dataset where we selected the **target variable ('Adj Close')**.
- Used **60 time steps**, so the model learns from the past 60 days to predict the next day's price.
- After applying the shape of X,y variables are,

X.shape = (9763, 60, 1)

- I. 9763 → Number of samples (data points available after sequence creation)
- II. 60 → Each sample contains 60 past days (time steps) as input
- III. 1 → Only one feature (Adj Close price)

y.shape = (9763,1)

- I. 9763 → The number of target values (corresponding next-day prices)

Train Test Split:

- Split the data using train_test_split: 80% for training and 20% for testing.

Train Shape: (7810, 60, 1), Test Shape: (1953, 60, 1)

- I. Total samples before splitting = 9763.
- II. Training set (80%) → 7810 sequences
- III. Testing set (20%) → 1953 sequences
- IV. Each sample contains → 60 past days as input
(time_steps = 60)
- V. Features per timestep → 1 (only Adj Close price).

Defining LSTM:

- Used a Sequential model with two LSTM layers consisting of 30 and 20 units respectively.
- Used 2 Dropout layers of 0.2, 0.2 to reduce over fitting.
- Used 2 Dense Layers where one is hidden layer and another one Output Layer.

Model Summary:

Model: "sequential_2"

Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 60, 30)	3,840
dropout_4 (Dropout)	(None, 60, 30)	0
lstm_4 (LSTM)	(None, 20)	4,080
dropout_5 (Dropout)	(None, 20)	0
dense_4 (Dense)	(None, 25)	525
dense_5 (Dense)	(None, 1)	26

Total params: 8,471 (33.09 KB)

Trainable params: 8,471 (33.09 KB)

Non-trainable params: 0 (0.00 B)

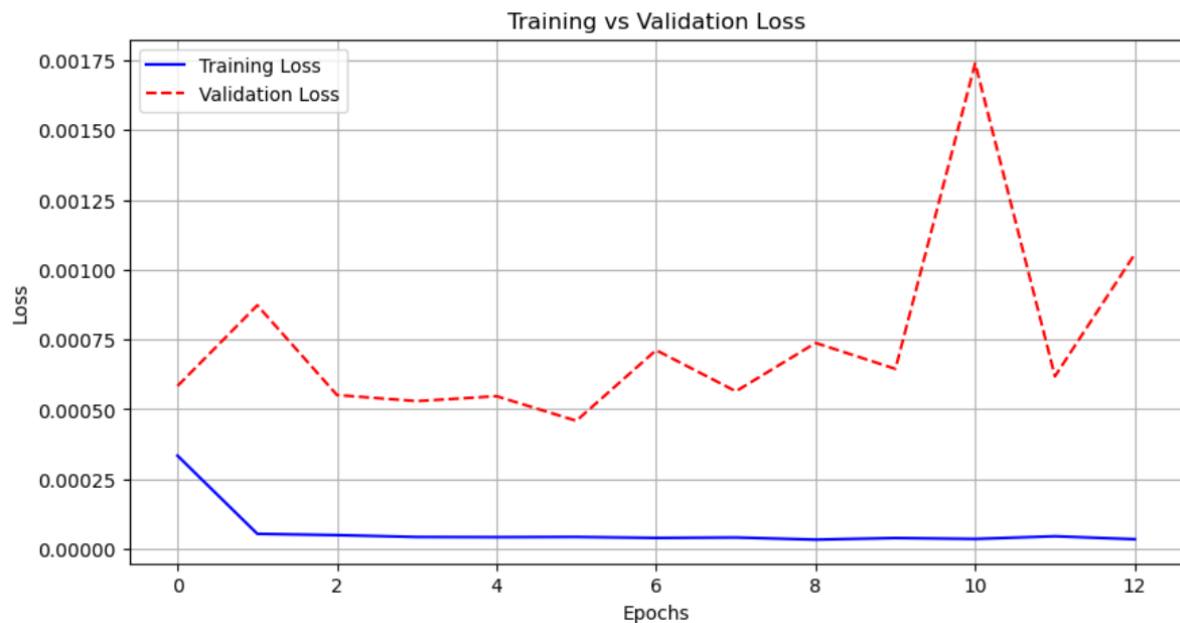
Compiling the Model:

- Compiled the model using the Adam optimizer with a learning rate of **0.0003**.
- The loss function used is Mean Squared Error (MSE).

Training the Model:

- Initially early stopping and model checkpoints are used, where early stopping prevents unnecessary learning if there is no improvement in val loss.
- Model Checkpoint is used to save the best model.
- Total Epochs is 50 in which model trained up-to 13 Epochs due to early stopping.

Training Vs Validation Loss:



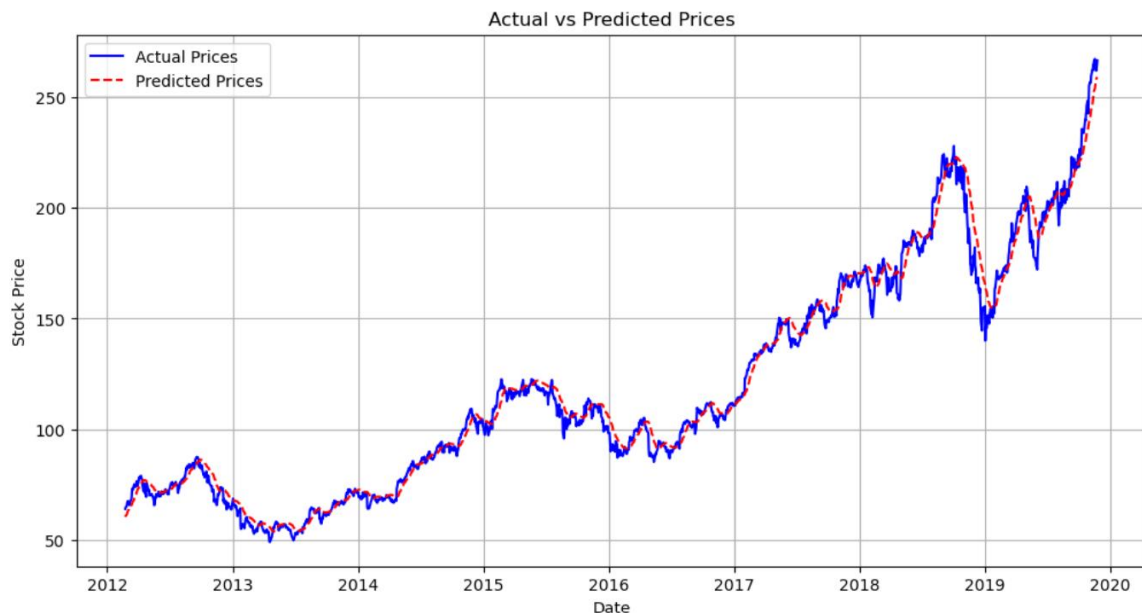
Predictions:

- Predictions are made on test data.
- Used inverse transformation to return predictions to original scale (undoing MinMax scaling).
- Calculated the Mean Squared Error.

Mean Squared Error (MSE): 32.68996681346626

- Lower Mean Squared Error indicates lower error, which means the predicted stock prices are closer to the actual stock prices.

Actual vs Predicted Prices Visualization:



Predictions of 1,5 and 10 days behavior:

1. Next Day stock price prediction:

1/1  0s 47ms/step

Predicted stock price for the next day: \$258.90182

2. Next 5 days stock price prediction:

1/1  0s 67ms/step

1/1  0s 41ms/step

1/1  0s 33ms/step

1/1  0s 33ms/step

1/1  0s 34ms/step

Predicted stock prices for the next 5 days: [258.90182, 259.44122, 259.85733, 260.17947, 260.43213]

3. Next 10 days stock price predictions:

1/1  0s 43ms/step

1/1  0s 35ms/step

1/1  0s 49ms/step

1/1  0s 35ms/step


1/1  0s 31ms/step

1/1  0s 35ms/step

1/1  0s 38ms/step

1/1  0s 38ms/step

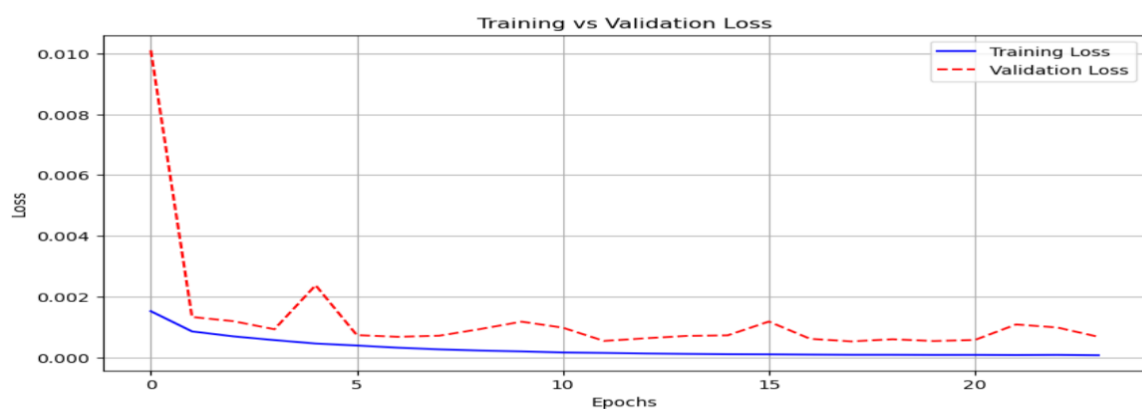
1/1  0s 44ms/step

1/1  0s 32ms/step

Predicted Stock prices for the next 10 days: [258.90182, 259.44122, 259.85733, 260.17947, 260.43213, 260.63406, 260.79886, 260.9366, 261.05475, 261.1587]

Difficulties Faced:

- Experimented with various dropout rates (0.3, 0.2, 0.4, 0.1) and finalized at 0.2 for both layers.
- Added another hidden dense layer
- Tested early stopping patience values of 5, 6, and 7.
- Tuned learning rates: tried 0.0005, 0.0003, 0.0002, and 0.0001.
- Below are the samples of training vs validation loss visuals when tried with the above usages.



- The above sample training vs validation loss is perfect but the Test_loss was very high so rejected the above.

Models Comparison:

	Actual Price	LSTM Predicted	SimpleRNN Predicted
Date			
2012-02-23	64.187332	60.589500	58.474442
2012-02-24	64.935638	61.051270	58.836296
2012-02-27	65.352058	61.514156	59.346649
2012-02-28	66.551537	61.976109	59.832184
2012-02-29	67.425362	62.455853	60.507729
2012-03-01	67.677681	62.957207	61.295155
2012-03-02	67.765923	63.467484	61.957336
2012-03-05	66.271889	63.973999	62.466251
2012-03-06	65.911377	64.427933	62.437740
2012-03-07	65.964828	64.825996	62.197220
2012-03-08	67.369415	65.175438	62.017994
2012-03-09	67.764725	65.516258	62.218758
2012-03-12	68.613670	65.850319	62.570160
2012-03-13	70.614899	66.192535	63.058891
2012-03-14	73.284874	66.581841	64.008858

- From the comparison, it is evident that the **LSTM** model performs **better** than the **SimpleRNN** in terms of prediction accuracy.