Week ML3 – Data Pre-Treatment & Baseline Models

H. Majeed, S. Bukhari, S. Gohar

November 24, 2025

1. Data Transformation

The raw dogecoin price series showed non-constant variance due to massive spike in the 2021. To stabilize this and remove the long term trend the following transformations were applied to the daily closing price:

- Log transformation to convert the absolute changes to log returns
- [-1, 1] min-max scaling to normalize to ensure input feature fall within a certain range and to stabilize the model convergence

2. Sequence Generation

The time series data was restructured into a format of (samples, timesteps, features) that is expected by Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM.

- Dataset was split into training and testing set as seen in Fig 1.
- A sliding window of 10 was used to create sequences. Each 10 past values serves as input and the 10+1 serves as target

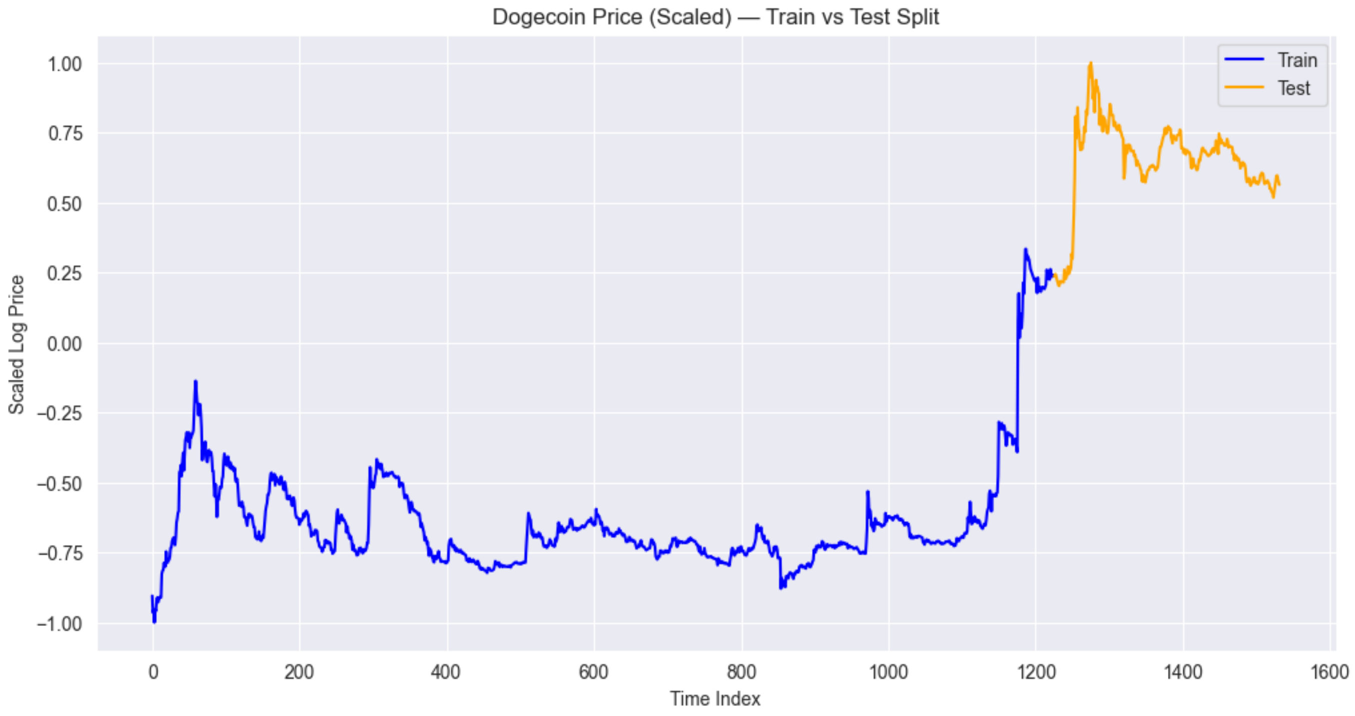

Figure 1: Train vs test split of Dogecoin data

3. RNN and LSTM Results

This part of the project involved training RNN and LSTM architectures after the baseline model. The goal was to determine if the models can learn the non linear relationships. The loss curve of the training Fig 2. shows fast initial convergence hinting that both models effectively learned the most dominant pattern in the data. The fact that LSTM’s loss quickly plateaus near the RNN’s is an indicator of that the market is highly random and noisy. LSTM’s core advantage of gating mechanism to capture long range dependencies is not being utilized because no strong long term signal exists. The predictive power is limited to the very recent look back window.

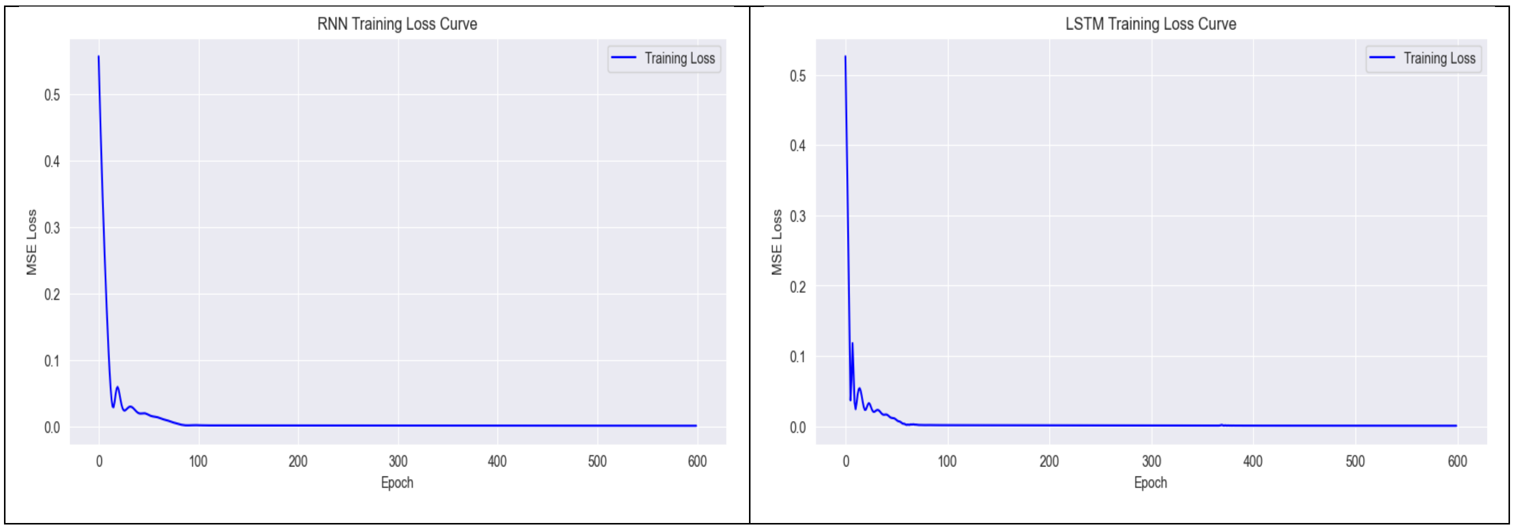

Figure 2: Loss curve comparison of RNN and LSTM

Both RNN (Fig 3.) and LSTM (Fig 4.) successfully generate a forecast that follows the general trend of the volatile Dogecoin price, especially when compared to the baseline Autoregression model which generated a flat line when asked to predict the price. Despite LSTM being a more advance method it is performing similar to RNN as its ability to solve the vanishing gradient problem and capture long term dependencies was offset by the noise dominance short term focus. Our model is predicting log returns, a highly noisy series which follows closely to the random walk hypothesis. In this scenario the prediction for tomorrows return is influenced by most recent 10 returns. The complex memory cell of LSTM is designed to remember steps for even 100 timesteps ago and this cell becomes irrelevant because market noise overrides any subtle long term signal. Since both RNN and LSTM perform well at short term dependencies their performance looks similar.

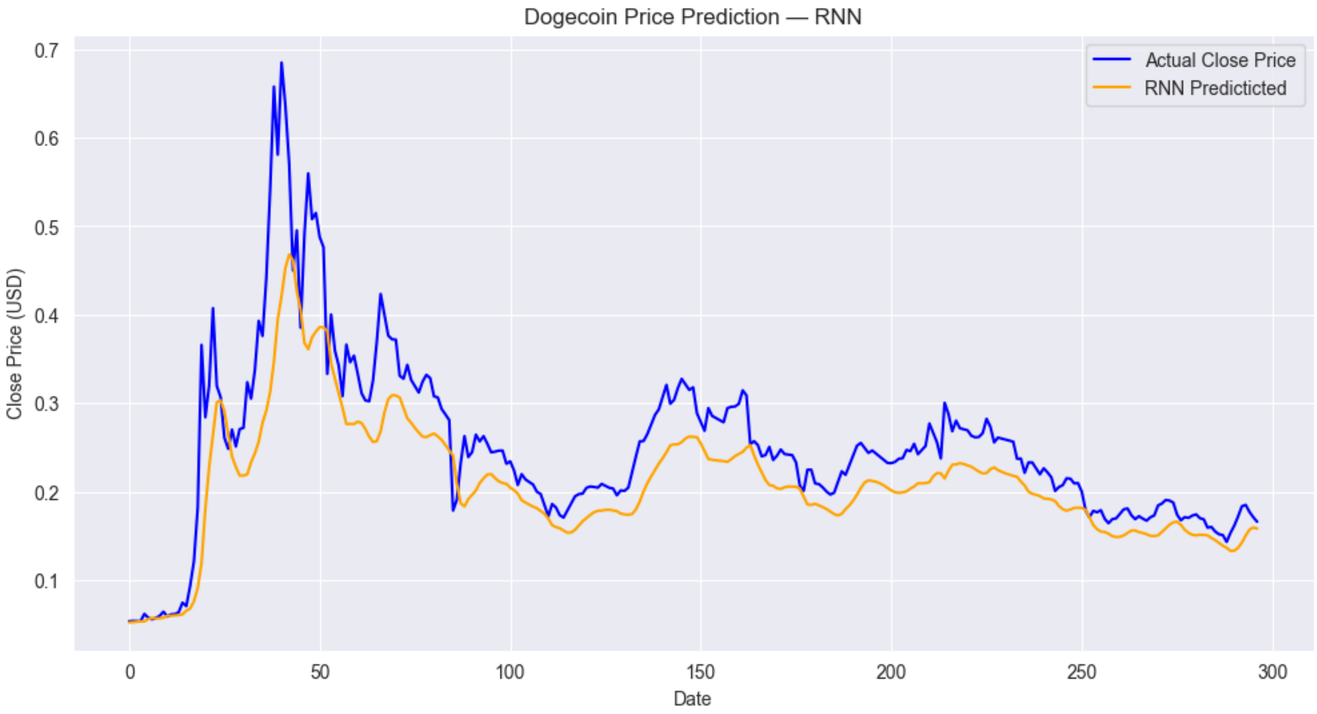

Figure 3: Actual vs predicted price using RNN

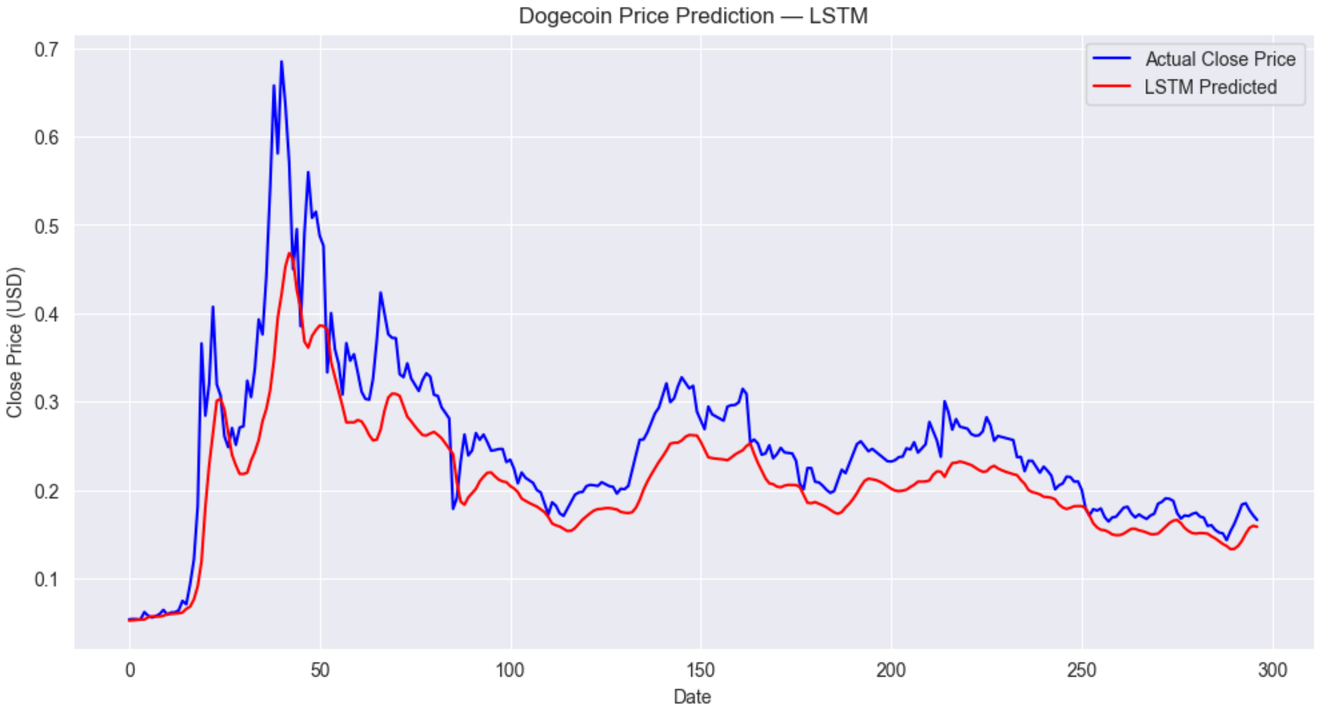

Figure 4: Actual vs predicted price using RNN

3. Model Improvement

The goal is to overcome the model's reliance on short term memory by providing external signals and increasing network capacity so we will be suggesting three improvements

3.1 Feature Engineering

To transform the model from univariate to multivariate we will be needing feature engineering more importantly features that impact price like other crypto movements such as bitcoin which shows what is the overall sentiment of the crypto market. Another feature that we can make use of is market volume which often acts as precursor to price movements.

3.2 Architecture Upgrade

To make use of the new features we will be needing a more advanced model architecture. We can make use of Stacked LSTM to increase model complexity or Bi-directional LSTM which can process the sequence in both forward and backward directions to enrich the context for each timestep.

3.3 Hyperparameter Tuning

Finally we tune our hyperparameters such as learning rate, number of neurons in hidden layer, adding dropout layer to prevent overfitting.