Task 5 Report

Naca Hitchman - 103072170

# Multistep problem



My function create\_sequences\_ms helps to solve the multistep prediction problem. This function allows predictions to be made for a sequence of closing prices of n\_steps\_ahead days into the future. The create\_sequences\_ms function takes three parameters: data, seq\_length, and n\_steps\_ahead. It creates two arrays, xs and ys, which store the input and output sequences for training the machine learning model. The function iterates over the data and appends subsequences of length seq\_length to xs, and then extracts another subsequence of length n\_steps\_ahead from the data for the y sequence, starting from the index immediately following the end of the x subsequence. This subsequence represents the target closing prices to be predicted.



I also have adjusted the code for actually outputting next k days’ stock prices. It starts by preparing the input data (real\_data) using the last available data from the test set. The shape of real\_data is adjusted to match the expected input shape of the model. Then, the model predicts the next k days’ stock prices using the prepared input data. The predicted prices are transformed back to their original scale using a column scaler. Finally, a loop is used to print each day’s predicted price.

# Multivariate problem



My function above helps to solves the simple multivariate prediction problem. It creates sequences considering other features of the same stock (including opening price, highest price, lowest price, closing price, adjusted closing price, and trading volume) as input for predicting the closing price of the company for a specified day in the future. The more important work comes from passing the data to it, as all the features of the data must be passed, and not just the Close column. Creation of the model must also be adjusted to account for this, including passing the feature count, and reshaping the model output to include the feature count.



As can be seen here, as the model prediction now outputs data with a shape that includes the feature count, we need to get the index of the column we want (which is ‘Close’ in this case) and use that to only get the data for that part of the prediction output.

# Combination

Combining the two is as simple as just including the necessary code for both systems as described above. The only code that needs to be adjusted to fit both systems is in the model creation, where the dense layer and output shape need to be adjusted.



As can be seen here, when the dense layer is added, we are also using Time Distributed to apply it to each time step, assisting with both the multi step and multi variate factors. The Lamba layer then must be added to ensure the model output is the correct shape.

# Results

Firstly, for the multi-step, we can the prediction successfully displays for the specified number of days in the future.



A black screen with white numbers

Description automatically generated

Moving on to multi-variate, the left is the result of my v0.4 code, and the right is the result of my current code. Neither are especially accurate. It may mean a more complex model is needed that would take longer to train to properly see results. Regardless, the code does fulfill the specifications of the task.

