COS30018 - Option B - Task 5: Machine Learning 2

Jiin Wen Tan 102846565

Resources:
https://www.analyticsvidhya.com/blog/2018/09/multivariate-time-series-guide-forecasting-modeling-python-codes/

https://link.springer.com/chapter/10.1007/11731139_89#:~:text=Multistep%2Dahead%20prediction%20is%20the,in%20the%20next%20time%20step.

https://www.kaggle.com/code/thibauthurson/stock-price-prediction-with-lstm-multi-step-lstm

Multistep Prediction

```python
def prepare_multistep_data(data, steps_in, steps_out):
    X, y = [], []
    for i in range(len(data) - steps_in - steps_out + 1):
        seq_in = data[i:i + steps_in]
        seq_out = data[i + steps_in:i + steps_in + steps_out]
        X.append(seq_in)
        y.append(seq_out)
    return np.array(X), np.array(y)
```

In the function above to achieve multistep prediction. First takes in data, steps_in and steps_out as parameters. Steps_in as how many days we set as an input and steps_out will be how many future days we want to predict as an output. Both sequences are then stored in x and y array. The for loop is to ensure that the function does not exceed the data boundaries and capture all possible input and outputs. Lastly, returning arrays x and y.

Multivariate Prediction

```python
def prepare_multivariate_data(data, features, steps_in):
    X = []
    for i in range(len(data) - steps_in):
        seq = data[i:i + steps_in][features].values
        X.append(seq)
    return np.array(X)
```

As we solved multistep predictions here comes multivariate problem the aim is to take in more than one features (default to be closing price) but now all of it. This function will iterates through dataset then extract consecutive sequences of length of steps_in for the specified features. Then these sequences will be stored in array x.

Combine multistep and multivariate prediction

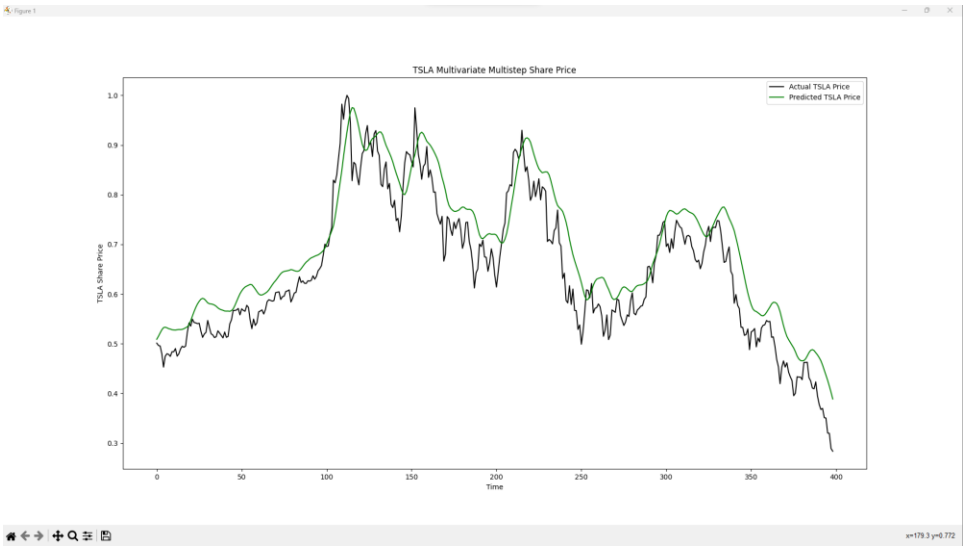
```python
def prepare_multivariate_multistep_data(data, features, steps_in, steps_out):
    X, y = [], []
    for i in range(len(data) - steps_in - steps_out + 1):
        # seq_in = data[i:i + steps_in][features].values
        seq_in = data.iloc[i:i + steps_in][features].values

        seq_out = data.iloc[i + steps_in:i + steps_in + steps_out]["Close"].values
        X.append(seq_in)
        y.append(seq_out)
    return np.array(X), np.array(y)
```

Pretty straightforward to merge both functions into one to achieve multivariate multistep at the same time.

Results of future 5 days closing prediction:

```
Predicted price for Day 1: 0.5198992490768433
Predicted price for Day 2: 0.4766560196876526
Predicted price for Day 3: 0.47895222902297974
Predicted price for Day 4: 0.47815367579460144
Predicted price for Day 5: 0.4667966365814209
```



^ I understand that the accuracy is way off the chart. But able to achieve multistep multivariate prediction.