Post-COVID19 Stock Crash Theoretical Model (RNN)

Chosen analysis period from 23rd March to 29th May (10 week period)
- Start date chosen based off multiple US and UK stocks from different industries – price action has some semblance of normality from the 23rd (which was also the first day after the weekend). Stock prices on the 18th and 19th were still largely effected by the crash in several industries.
- End date chosen as was the latest point which could be analysed at the time of creation, and it also concluded the week.

Chosen stock: American Airlines (AAL)
- AAL was chosen due to its high volume and volatility (but still within reason) within the coronavirus period, it was also a stock I actively traded during this time and as such has high familiarity. This would allow for potential deeper insights to be realised.
- The target stock can also be easily switched out for another and the same analysis can be rerun

– IMPORTANT – Note on resolution and training time
The model was initially tested on a 1-minute resolution period, NOT a 15-minute period – as such there were about 43888 different time points, each with 5 features; open, close, high, low, volume. I was not able to fit/train this model effectively as a single epoch would have taken 18 hours on the current hardware, even Google Colab notebooks struggled with this.
- Like other aspects of this model, this change is extremely easy to implement, only the retrieval query needs to be edited. Resolution list: 1, 5, 15, 30, 60, D, W, M

Data

- Dates needed to be in UNIX format to retrieve prices from the API
- Data consists of 15-minute intervals (rarely, the opening 15m and the immediate following 15m period might vary slightly)
- Opted for unadjusted data to get the most direct figures to base the algorithm on, furthermore many dividends etc. were frozen during this time so adjusted data would likely be similar to unadjusted (retrieving adjusted data can easily be changed in the request)

RNN Model

The RNN model made use of EarlyStopping(patience=3) and trained on the 5 features. It was quite difficult to test different configurations and hyperparameters with the model as, even with a reduced resolution of 15 minutes, training time was significant.
Training was conducted through Google Colab as it was slightly quicker than my local machine – training time took about 30 minutes.

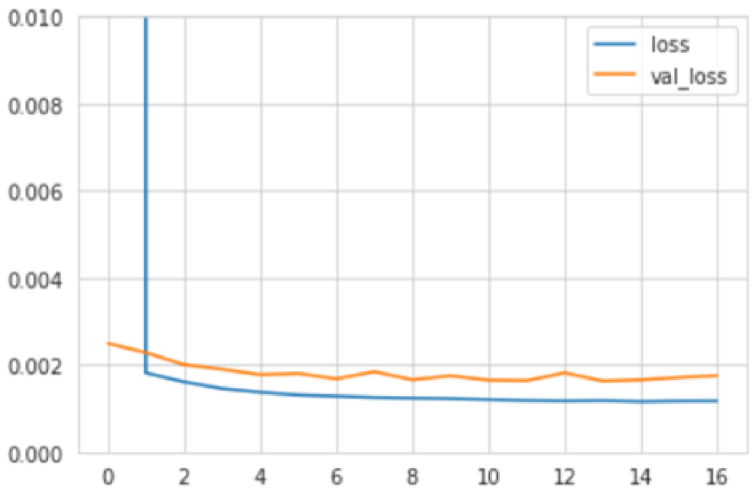
A test size of 20% was used, with a generator length of 12% of the entire dataframe; interestingly, slightly tighter figures like 18%/14% returned null losses and val_losses in the metrics and as such were not usable.

RELU activation was used on the 32 neuron LSTM layer and the ADAM optimiser was used for compiling the model. These decisions were made based on previous RNN testing where I found the RELU/ADAM model to be superior to other combinations of RELU/ADAM/TANH/RSMPROP models.

Example Model Metrics – 32 Neurons, 15m resolution, all 5 features, EarlyStopping(patience=3)

```
[43] metrics = pd.DataFrame(model.history.history)
     metrics.plot()
     plt.ylim(0,0.01)
```

```
(0.0, 0.01)
```

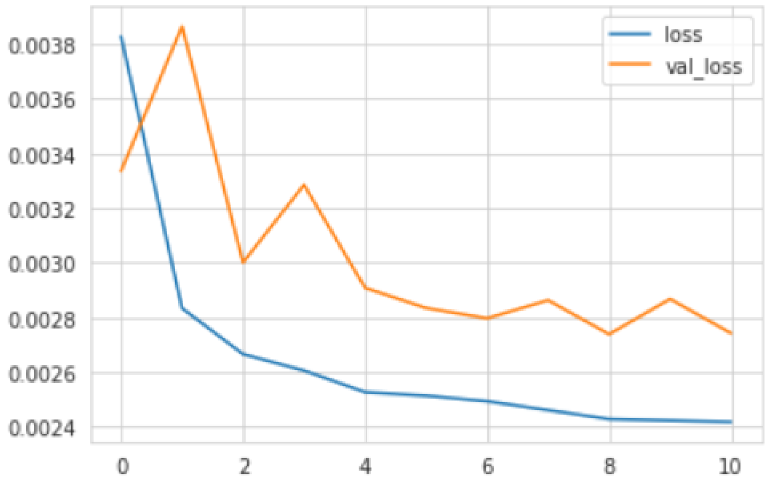


Example Model Metrics – 30 Neurons, 15m resolution, all 2 features, EarlyStopping(patience=2)

```
[37] metrics = pd.DataFrame(model.history.history)
     metrics.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f753007f4e0>
```

No



reliable or applicable findings resulted from this simple RNN. Whether in price, or volume.

The model was not designed for actual predictions but simply shows the basic methodology of using an RNN to forecast stocks. For a more tangible and interpretable result, non-deep-learning methods should be employed, such as SARIMAX.

Improving the RNN for REAL usage

Increase Features
On the other hand, to effectively use an RNN to predict stock price action, we would need to retrieve (and maybe engineer) more features (like what was done for my other project, UK Population Forecast). This would give the network more varied points to learn from to develop a robust and dynamic network.

Set Time Period
It may also be beneficial to change the scope of the time period, for example, specifically training the model for intraday price movements rather than weekly periods as this project did. Larger weekly periods are also possible but would need a larger set of data than just during the coronavirus period.

Select a Stable Stock/Time Period
Something key to note is that the RNN will do better if it can solve some sort of pattern within a more stable period and with a more stable stock – where fluctuations in prices have some resemblance of a cyclical nature. One reason for choosing the coronavirus period for this project was to see if anything at all could be predicted using a straightforward RNN – nothing at all was found to be useful.

Adjust Hyperparameters (And use better hardware)
Retraining the initial model using different hyperparameters, as is usual for refining a neural network, would increase the chances of coming across a more applicable model – although in order to do this, especially with a Timeseries of many features, significantly better equipment would need to be used. Hardware specifically designed for processing of this kind, along with the relevant software to be configured to use the hardware.