Using LSTM Network to predict Stock Prices
Franklin Jia - franky3@Stanford.edu

### Introduction
- **Stock Price Prediction** is a difficult task to tackle given only publicly available data.
- Predicting stock prices is best achieved by using a LSTM (Long Short Term Memory) because of its ability to remember information in the past.
- My goal is to build an LSTM network that can predict the prices of the following day, as well as the general tendency for a specified window of time.

### Data Preparation
- **Closing Prices** were extracted from the the csv file using pandas.
- Split the dataset into windows of a specified size, with the last value representing the price of the next day to be predicted.
- Normalize the dataset by dividing prices within a window by the first price in that window and subtracting 1.

### Hyperparameter Tuning

#### Hyperparameter Tuning
- Tuned batch size, number of hidden units in layer, timesteps, number of layers, and number of epochs.

#### Results (See Tables in Hyperparameter Results)
- Increasing batch size had little to no effect on the RMSE so chose larger batch size to decrease training time.
- Increasing number of hidden units decreased RMSE dramatically in the beginning, so chose 30 to balance training time and complexity when RMSE stabilized.
- Increasing timestep led to increasing RMSE, but increase was marginal, therefore intuitively larger timestep should enable algorithm to learn more about how past affects future.
- Number of layers didn't affect the RMSE much so chose a more complex network with an RMSE that was still reasonable (3 LSTM layers).
- Number of epochs (50) chosen to balance RMSE and runtime.

### Model

#### Setup:
Coded in Python and used Keras with Tensorflow backend to build the model.

#### Model:
- Linear stack of layers: 3 LSTM layers of 30 units each, fed into a Dense layer with linear activation.
- Minimize Root Mean Square Error and used RMSProp as optimizer.
- LSTM cell:

<table>
<thead>
<tr>
<th>Forget</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- LSTM cell has forget, input, and output gates which allow it to better capture long-term dependencies than standard RNN's.

#### Prediction:
- **Day-by-Day** takes the test examples and predicts the next price and strings those predictions together.
- **Whole Sequence** starts with the first window in test set, predicts next price, then cuts out the oldest price in window and appends the predicted price and predicts the next price on this new window (effectively shift by 1 using predicted data).
- **Tendency** performs the same process as Whole Sequence but only for a set number of days n. After n steps, it shifts forward by n and takes that window from test set and repeats.

### Model Evaluation

#### Day-by-Day Prediction
- The Day-by-Day Prediction scheme seems to be able to do a good job predicting the price at next day.
- Predict Whole Sequence scheme looks like it has the general idea correct, but predicts at a scale that is much larger than actual movement.
- Next-N-Days scheme seems to be predicting trends incorrectly, predicting downsizing where there should be upsizing.

#### Whole Sequence Prediction
- The Next-N-Days Prediction clearly still needs work as it is unable to predict the trends of the stock price correctly over the time period. This may be a result of compound errors from using predictions from previous timesteps in the window for next timestep.

### Conclusion and Future Work
- Both the Day-by-Day and Whole Sequence predictions proved to be rather successful at predicting the movement of the stock price.
- The Next-N-Days prediction clearly still needs work as it is unable to predict the trends of the stock price correctly over the time period. This may be a result of compound errors from using predictions from previous timesteps in the window for next timestep.

For future work, I'd like to focus on:
- Fixing Next-N-Days Prediction
- Building another network to predict stock price movement based on news articles and integrating that network with this one to see the results of predictions.
- Making a framework to utilize my model real-time.