Stock Price Predicting with LSTM using Keras

--Yiqiao Chen

Neural Networks have been demonstrated to be very powerful in solving the real-world problems like imaging and natural language processing. Interested in stock price movements, I am therefore motivated and curious about employing nerual nets to approach financial time series problems of its kind.

Unlike common regression predictions, time series problems are more difficult due to sequence dependence within the input data variables. Among various neural nets, Long Short-Term Memory (***LSTM***), a type of recurrent neural network, has been proven to take care of complex time series predictions. So I implement LSTM to Amazon's stock prices (***AMZN***) using ***Keras***.

1 Load Data

Get up to five-year daily historical stock prices & volumes data from http://www.nasdaq.com/quotes/historical-quotes.aspx (http://www.nasdaq.com/quotes/historical-quotes.aspx).

Datasets consist of following columns:

- **date** Time stamp of when data was collected, being used as index column;
- **volume** The number of shares traded in AMZN during a given trading day;
- **open** The price at AMZN upon the opening of an exchange on a given trading day;
- **high** The highest price at which AMZN traded during the course of the day;
- **low** The lowest price at which AMZN traded during the course of the day;
- **close** The final price at which AMZN traded during the course of the day; it is also selected as our target data.

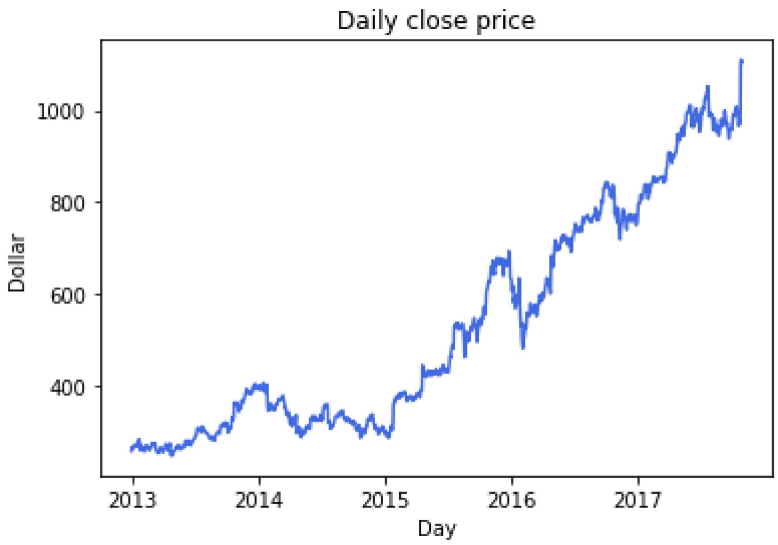
Plus, add a new feature based on the columns above:

- **pct** The percentage of change of close price compared to previous trading day.

```
Historical stock price data for 1217 days
from 2013-01-03 to 2017-10-31
```

date	volume	open	high	low	pct	close
2013-01-03	2750470	257.27	260.880	256.370	0.004547	258.4800
2013-01-04	1874011	257.58	259.800	256.650	0.002592	259.1500
2013-01-07	4909389	262.97	269.725	262.670	0.035922	268.4592
2013-01-08	3008773	267.07	268.980	263.567	-0.007745	266.3800
2013-01-09	2262402	268.17	269.500	265.401	-0.000113	266.3500

Visualize the five-year daily close price data



2 Data Preprocessing

2.1 Scale features

The input and output data that go into model need to be scaled. Use preprocessing.MinMaxScaler() function in scikit-learn library to scale data to the range of 0-1.

Scaled data

date	volume	open	high	low	pct	close
2013-01-03	0.072845	0.009685	0.009139	0.012416	0.455738	0.011882
2013-01-04	0.034326	0.010046	0.007898	0.012743	0.447958	0.012659
2013-01-07	0.167727	0.016313	0.019308	0.019781	0.580597	0.023451
2013-01-08	0.084197	0.021080	0.018451	0.020830	0.406821	0.021041
2013-01-09	0.051395	0.022359	0.019049	0.022974	0.437194	0.021006

2.2 Split data to training set and test set

Split data into training (60%) and test sets (40%).

```
Training set: 712 obs
Test set: 474 obs
```

3 Modeling

Now predict the close price for day m based on data observed in the past 30 days $\{m-30, m-29, \ldots m-1\}$.

3.1 Define Network

Define a **Sequential Model** and add:

- input layer with dimension (30, 6);
- two LSTM layers with 256 neurons;
- one hidden layers with 32 neurons with 'Relu';
- one linear output layer.

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 30, 256)	269312
dropout_1 (Dropout)	(None, 30, 256)	0
lstm_2 (LSTM)	(None, 256)	525312
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 32)	8224
dense_2 (Dense)	(None, 1)	33

Total params: 802,881
Trainable params: 802,881
Non-trainable params: 0

3.2 Compile Network

Before training, configure the learning process by specifying:

- **Optimizer** to be 'adam';
- **Loss function** to be 'mse';
- **Evaluation metric** to be 'accuracy'.

3.3 Train Network

Fit the model to training data to learn the parameters.

```
<keras.callbacks.History at 0x1c908adb048>
```

3.4 Evaluate Network

Evaluate the fitted model on test set.

```
474/474 [==============================] - 1s 1ms/step
mean square error =  0.0180481404129
```

4. Results

The model shows that LSTM can capture the pattern of Amazon's stock prices. However, there are some gaps between predicted and true movements. We can certainly improve the performance by further tuning the model, but the assumption of the stable status in stock market mechanism is too idealistic to return a more accurate result.

Since investor behavior is highly impacted by various factors, the future idea is to add sentiment analysis to better understand investors' pyschology. Therefore, we can capture more price catalysts in the market and achieve a better prediction with a holistic view.