Trent Leslie
12/2021

Capstone Three: LSTM Neural Network Candlestick Forecasting for Short-term Trading Opportunities

Problem Statement

Given that stock prices can be modeled by a random walk, forecasting prices may seem to be an exercise in futility. However, using neural networks to identify patterns in OHLC data may provide some insight for potential trading opportunities. In trading, a popular approach in technical analysis is the use of candlestick patterns to identify potential reversals or continuation in price changes over time. This relies on price over time reflecting collective psychology that tends to repeat itself and the tendency for prices to regress to a longer-term mean. Further, incorporating information other than price itself, such as volume, moving averages, and RSI, may yield more reliable models in practice as traders and algorithms rely on these for making decisions. In conjunction with some trading rules, such as avoiding shorts on indices (since they tend to go up over time) and stop-loss orders based on volume profile, a model or set of models may provide an edge to grow an account over time.

To reduce risk and maximize returns on short-term (1-5 trading days) trades, what are the expected open, high, low, and close (OHLC) prices for a given index, stock, ETF, forex, or futures contract tomorrow? What are the expected OHLC prices for the following week? What is the most effective rolling window size for training data?

Data Wrangling

The primary source for these datasets is the Alpha Vantage API, a free API for a wide variety of historical market data.

In general, the data collected for modeling was the following:

1) Open, High, Low, Close (OHLC) data for 21 ETF ticker symbols
2) Volume data for the same 21 ETF ticker symbols
3) 21-day exponential moving average (calculated on close)
4) 50-day simple moving average (calculated on close)
5) Relative Strength Index (14-day)

The 21 ETFs were chosen for their large volume of trading, representation of broad sectors of the market, and data quality (full data sets and statistical distributions resembling those of the rest of the group). This approach intended to avoid as much market noise of individual stocks as possible and provide as much signal to underlying price movement relationships as possible. The ETF data collected came from the following ETFs:

Trent Leslie
12/2021

1) SPY: SPDR S&P 500 ETF Trust
2) QQQ: Invesco QQQ Trust (Nasdaq)
3) XLF: Financial Select Sector SPDR Fund
4) EEM: iShares MSCI Emerging Markets ETF
5) XLE: Energy Select Sector SPDR Fund
6) SLV: iShares Silver Trust
7) FXI: iShares China Large-Cap ETF
8) GDX: VanEck Gold Miners ETF
9) EFA: iShares Core MSCI EAFE ETF
10) TLT: iShares 20+ Year Treasury Bond ETF
11) LQD: iShares iBoxx $ Investment Grade Corporate Bond ETF
12) XLU: Utilities Select Sector SPDR Fund
13) XLV: Health Care Select Sector SPDR Fund
14) XLI: Industrial Select Sector SPDR Fund
15) IEMG: iShares Core MSCI Emerging Markets ETF
16) VWO: Vanguard FTSE Emerging Markets ETF
17) XLK: Technology Select Sector SPDR Fund
18) IEF: iShares 7-10 Year Treasury Bond ETF
19) XLB: Materials Select Sector SPDR Fund
20) JETS: U.S. Global Jets ETF (airlines)
21) BND: Vanguard Total Bond Market ETF

Exploratory Data Analysis & Preprocessing

Because the preprocessing and exploratory data analysis went together in this project, these two are combined into a section and training is left to its own section below.

Table 1 below summarizes the standardization approach for each feature from each ticker and window size. These were done for each of the 21 ticker symbols above within each window size (10, 20, 30, 40, 80, 160, 320, and 640 periods). For each window size, the 21 ticker symbol data sets that were independently standardized. After standardization, each window was then flattened to one row, such that window size then determined the number of feature columns (window size * 8).

These flattened data sets were then concatenated into a combined data set. Each window size has its own, independently calculated combined data set that was calculated based on that window size. Each data set contains (window * 8) feature columns and four target columns (open, high, low, close). For splitting into training and test data sets, the shuffle argument was used to ensure random assortment of each ticker symbol into the separate training and test data sets.

Trent Leslie
12/2021

Table 1: Feature and Preprocessing Summary List

Column Name	Process
open	absolute dollar value -> percent change -> sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate
high	absolute dollar value -> percent change -> sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate
low	absolute dollar value -> percent change -> sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate
close	absolute dollar value -> percent change -> sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate
volume	absolute value -> percent change -> sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate
ema (exponential moving average)	percent of close -> sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate
sma (simple moving average)	percent of close -> sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate
rsi (relative strength index)	sliding window generation -> rolling z-score (standardization based on window size) -> flatten -> concatenate

The only exploratory analysis undertaken for this project was to evaluate the distributions of z-score descriptive statistics for the 10-period window size. The goal of this was to grow the dataset via concatenation with data of similar structure according to z-score, and consensus among the mean and standard deviations of z-scores suggests such similar data structure. As summarized above, the rolling z-scores were calculated for each window throughout each of the ticker datasets. For each ticker in the 10-period window, the mean and standard deviation of the z-scores for each feature was extracted. As illustrated in Figure 1, a histogram for the means across the ticker data sets was generated for the open, high, low, and close features.

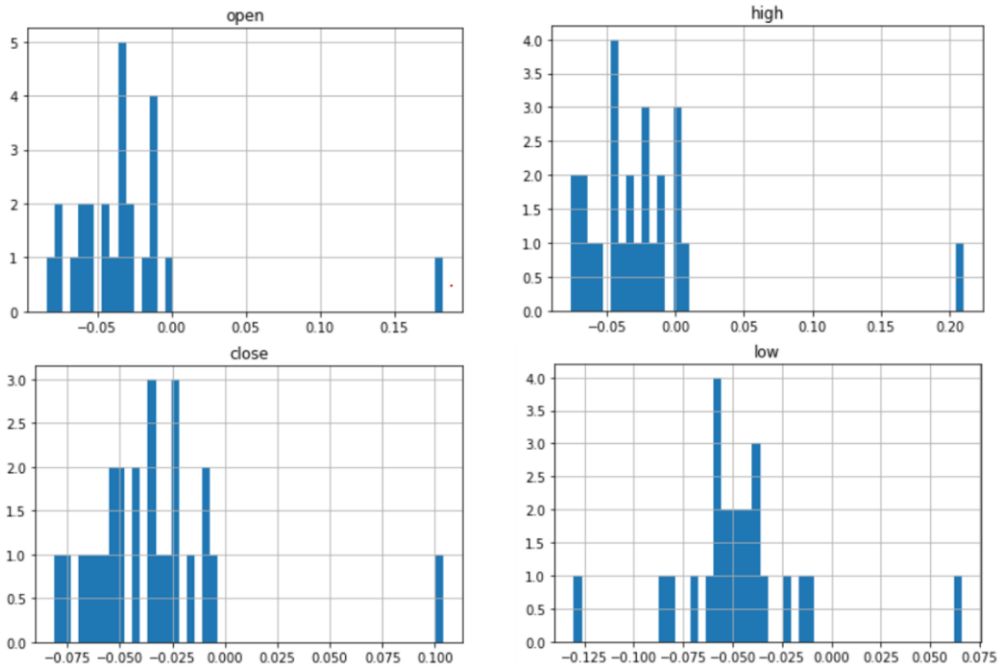

Figure 1: Histogram distributions of mean z-scores for the respective feature in the 10-period window

Prior to arriving at the 21 ETFs mentioned above, four ETFs were removed from the starting list. One was removed for being an outlier to the upside, as indicated in all subfigures above. Another was removed for being an outlier to the downside, as illustrated in the low subfigure above. Finally, two were removed due to having zero volume values, which created infinite values in the percent change calculation. These four ETFs and relevant values are shown in Table 1 below.

This resulted in a training data set of ~100,000 rows for each of the eight window sizes, of which each was a concatenation of the 21 ETF flattened data sets.

Table 1: A summary of the removed ETFs and their relevant statistics

ticker	stat	open	high	low	close	volume
EWZ	mean	-0.035341987	-0.022547611	-0.041337936	-0.035761034	inf
EWZ	std	1.318922066	1.296249446	1.311434566	1.269684114	
HYG	mean	-0.075503843	-0.069374342	-0.130310442	-0.081008299	0.108681252
HYG	std	1.377978434	1.42122522	1.4553079	1.379505735	1.564697531
IEF	mean	-0.01353213	0.003001705	-0.021122435	-0.009079921	inf
IEF	std	1.285227133	1.297855472	1.288484378	1.254383555	
JNK	mean	0.182613617	0.209941282	0.065949746	0.103816158	0.126315129
JNK	std	16.25910553	18.23264211	11.72257763	11.77547138	1.638226284

Training

Training was conducted on a local laptop environment running Tensorflow 2.1.0 with Python 3.6.6. Tensorboard 2.1.1 was used for evaluating model performance. Further, the epoch loss and epoch mean absolute error used for model performance were evaluated only on target close values (not target open, high, or low values).

To reduce loss and mean absolute error on the validation set, training involved using this paper as a rough guide and working through the following general steps:

1) Starting with a 10-period window size, experiment with epoch count, activation functions, layers, and unit count.
2) Epoch count did little past 25 epochs. Linear, elu, selu, and tanh activation functions performed the best, with none of them clear winners. Layers offered little improvement with more than three layers, and adding orders of magnitude more layers was actually a detriment. Unit counts moving from 40 to 1000 offered a great improvement, and 1,500 units offered a marginal improvement. However, computer memory issues tended to happen more at 1,500 units and above.
3) Increasing window size also had a drastic and positive impact. Marginal gains were made up to a window size of 320, but gains leveled off with the 640-period window size.

These lessons are illustrated in Figure 2 below. Two sets of four models were saved – linear, elu, selu, and tanh activation models for both 20-period and 320-period window sizes. While the 320-period

models may perform marginally better than the 20-period models, it also encompasses a window that spans 15 months. The speculation is that the shorter 20-period model may specialize on the noisier short-term movements while the longer 320-period model may incorporate less noise and more seasonality. In practice, they may complement each other.

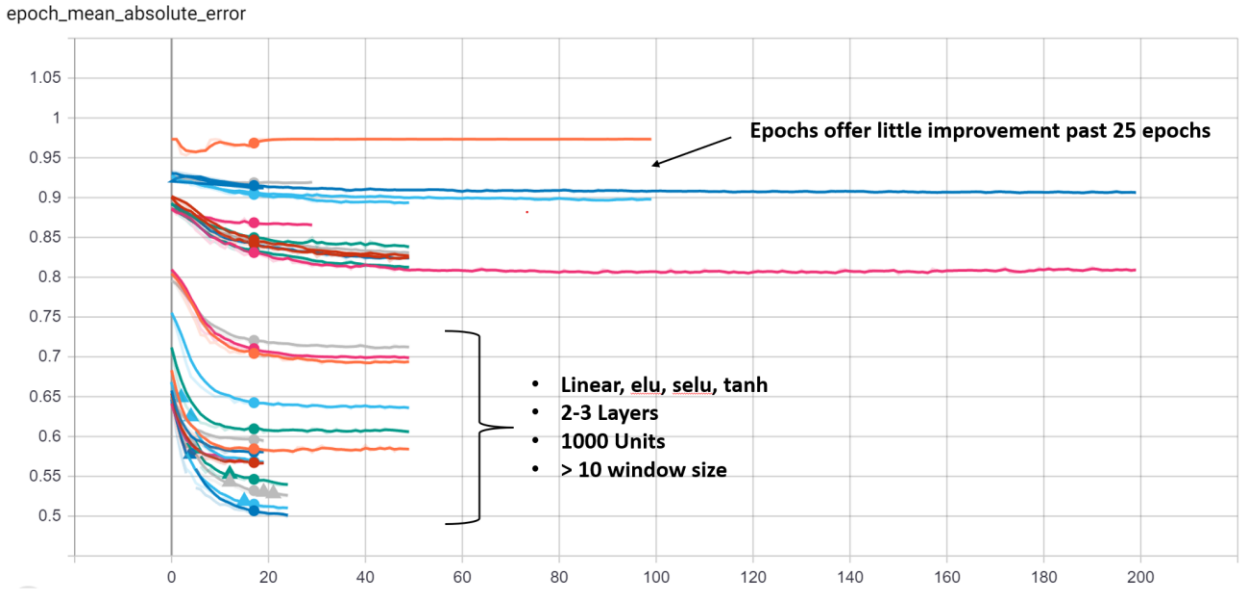

Figure 2: Mean Absolute Error over Epoch Training

The eight selected models were then used to predict the last five trading days of activity on SPY, the S&P 500 index ETF. Unfortunately, the training data was accidentally updated, and these five days were included in the training and/or test data sets. At any rate, the predicted vs. actual plots for open, high, low, and close z-score values with all eight models are shown in Figure 3 below. The models are named as "[activation function]_[window size]".

One thing to note in these figures is that the actual z-scores for the two window sizes are different. As mentioned above, each window size has its own independently calculated data sets, and this is made apparent with the actual values below. Some are close, but the vertical lines indicate the ten actual values for each price over the five trading days (2 window sizes over 5 trading days is 10 actual values).

Another standout is the linear_20 model performance on the low z-scores. While most models seem to be in the neighborhood at higher z-scores, linear particularly stands out at the lower z-scores. It could just be luck or the fact that the trading day was a part of the training data set, but it correctly called the rout on 11/26/21. It not only suspected the down closing price, but the gap down open price as well. The eerie part about that is that it would have been based on the previous month's technical data, without any knowledge of the story tied to that trading day's performance - news of the omicron covid variant that came out when markets were closed on Thanksgiving Day. Regardless, it will be interesting to see if this model can provide any warning of any major selloff days in the future.

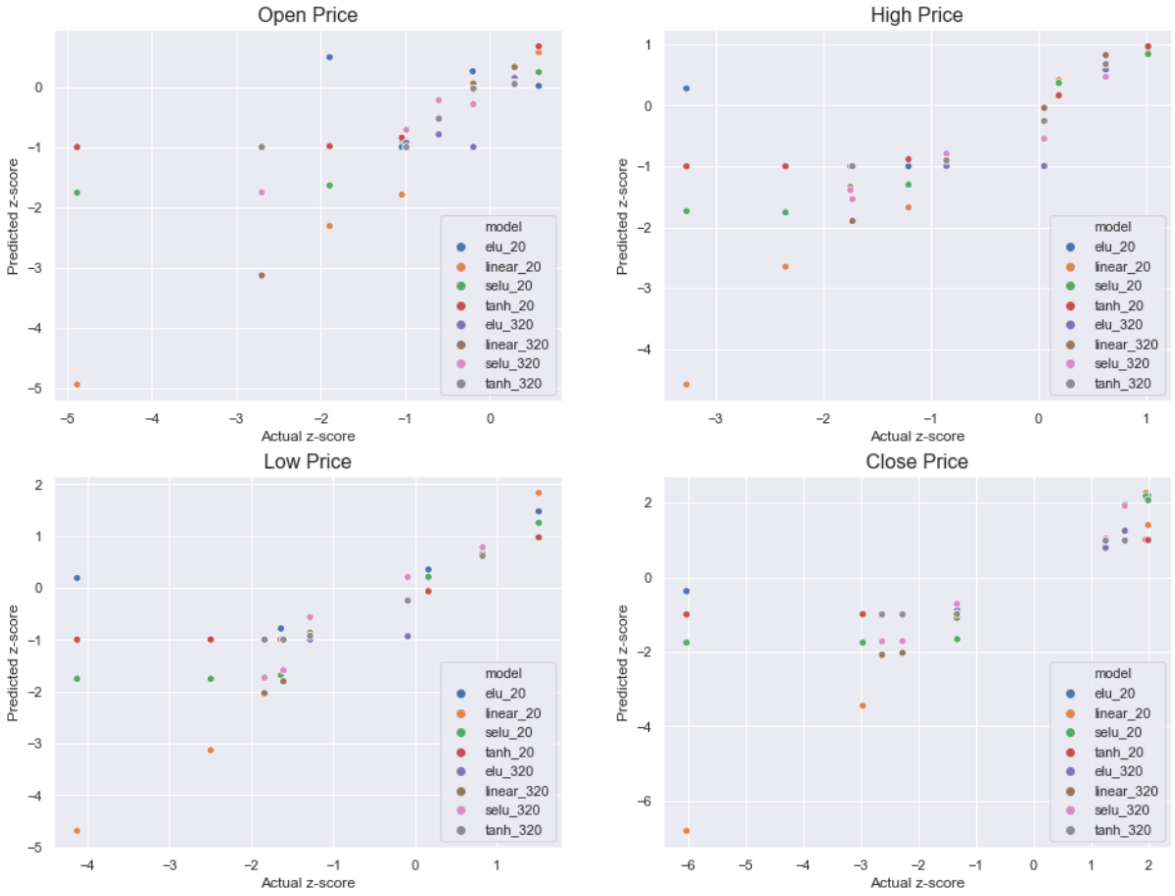

Figure 3: Predicted vs. Actual z-scores for the Last Five Trading Days (11/26/21-12/2/21)

The final check was to see how this looked in real life – that is, with real numbers instead of z-scores. To accomplish this, the mean and standard deviation from the final sliding window used to predict the next day's prices was used to back-calculate the absolute dollar values from the z-scores. The actual values were then used to calculate percent error based on the absolute dollar value. Table 2 below demonstrates that this was only done for the most recent trading day on SPY, as it was the first trading day outside the training and test data sets.

While some are remarkably accurate, such as linear-20 on the open, tan-20 on the high, and elu-320, others far exceed a typical trading day's range. It will be interesting moving forward to see if the remarkable performance is random across prices and models or if the models above consistently perform well at the respective prices over time. At the time of this writing, the linear-20 model is forecasting a 0.55% gap up in the SPY tomorrow morning (12/6/2021), and the S&P 500 futures (ES) are currently up 0.45%. Regardless of this promising status, the placement of any market orders will be delayed until the performance of these models is better understood.

Table 2: Percent Error on Absolute Values for SPY on 12/3/2021

SPY 12/3/2021	elu-20	linear-20	selu-20	tanh-20	elu-320	linear-320	selu-320	tanh-320
Forecast Open	$ 454.67	$ 459.04	$ 456.76	$ 453.62	$ 453.08	$ 452.84	$ 453.27	$ 453.52
Forecast High	$ 458.04	$ 461.55	$ 461.49	$ 460.26	$ 459.40	$ 458.70	$ 459.28	$ 459.95
Forecast Low	$ 450.85	$ 457.55	$ 455.50	$ 451.75	$ 448.85	$ 450.43	$ 451.81	$ 452.16
Forecast Close	$ 455.68	$ 459.39	$ 458.57	$ 457.24	$ 454.47	$ 454.32	$ 456.64	$ 456.84
Actual Open	$ 459.17	$ 459.17	$ 459.17	$ 459.17	$ 459.17	$ 459.17	$ 459.17	$ 459.17
Actual High	$ 460.30	$ 460.30	$ 460.30	$ 460.30	$ 460.30	$ 460.30	$ 460.30	$ 460.30
Actual Low	$ 448.92	$ 448.92	$ 448.92	$ 448.92	$ 448.92	$ 448.92	$ 448.92	$ 448.92
Actual Close	$ 453.42	$ 453.42	$ 453.42	$ 453.42	$ 453.42	$ 453.42	$ 453.42	$ 453.42
% Error Open	0.98%	0.03%	0.52%	1.21%	1.33%	1.38%	1.29%	1.23%
% Error High	0.49%	0.27%	0.26%	0.01%	0.20%	0.35%	0.22%	0.08%
% Error Low	0.43%	1.92%	1.47%	0.63%	0.02%	0.34%	0.64%	0.72%
% Error Close	0.50%	1.32%	1.14%	0.84%	0.23%	0.20%	0.71%	0.75%

Future Work

Future work for this project breaks down into a few categories, namely:

1) Expanded use of the current models: Not only more tickers, but datasets involving different timeframes, such as weekly data, and different asset classes, such as forex and cryptocurrency.
2) Development of new models: a) Using the same training data developed in this project to train a model to forecast the next week's OHLC candle. It would be identical to these models but would be trained to the next week's OHLC data instead of the next day's OHLC data. The speculation is that the next day may be trying to model more noise, whereas the next week may provide more signal and reduce mean absolute error. b) It would also be interesting to add some one-hot encoding data to the training data set. Namely, incorporate one-hot encoding to tie some values to important sectors, such as treasuries, corporate bonds, and precious metals via the ETF price data already in the dataset. c) Incorporating more and/or different technical indicators to the model.
3) Move more intensive training to the cloud: It would be interesting to have more memory to explore adding more units to each layer, particularly orders of magnitude higher.

Supplemental Information

The h5 model files are too large for GitHub, so they are located on Google Drive here.

The training data sets are also too large for GitHub, so they are located here.

Each trained model has its own Jupyter notebook and can be found here.