[Sign In](https://medium.com/m/signin?operation=login&redirect=https%3A%2F%2Ftowardsdatascience.com%2Fgetting-rich-quick-with-machine-learning-and-stock-market-predictions-696802da94fe&source=post_page---three_column_layout_nav-----------------------three_column_layout_nav-----------)

[](https://medium.com/m/signin?operation=login&redirect=https%3A%2F%2Ftowardsdatascience.com%2Fgetting-rich-quick-with-machine-learning-and-stock-market-predictions-696802da94fe&source=post_page---three_column_layout_nav-----------------------lo_home_nav-----------)

[](https://towardsdatascience.com/?source=post_page-----696802da94fe--------------------------------)

Published in

[Towards Data Science](https://towardsdatascience.com/?source=post_page-----696802da94fe--------------------------------)

You have **2** free member-only stories left this month.

[Sign up for Medium and get an extra one](https://medium.com/m/signin?operation=register&redirect=https%3A%2F%2Ftowardsdatascience.com%2Fgetting-rich-quick-with-machine-learning-and-stock-market-predictions-696802da94fe&source=-----696802da94fe---------------------smart_meter-----------)

[](https://medium.com/@yacoub.ahmedy?source=post_page-----696802da94fe--------------------------------)

[Yacoub Ahmed](https://medium.com/@yacoub.ahmedy?source=post_page-----696802da94fe--------------------------------)

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F704610cc196f&operation=register&redirect=https%3A%2F%2Ftowardsdatascience.com%2Fgetting-rich-quick-with-machine-learning-and-stock-market-predictions-696802da94fe&user=Yacoub+Ahmed&userId=704610cc196f&source=post_page-704610cc196f----696802da94fe---------------------follow_byline-----------)

Oct 11, 2019

·

12 min read

·

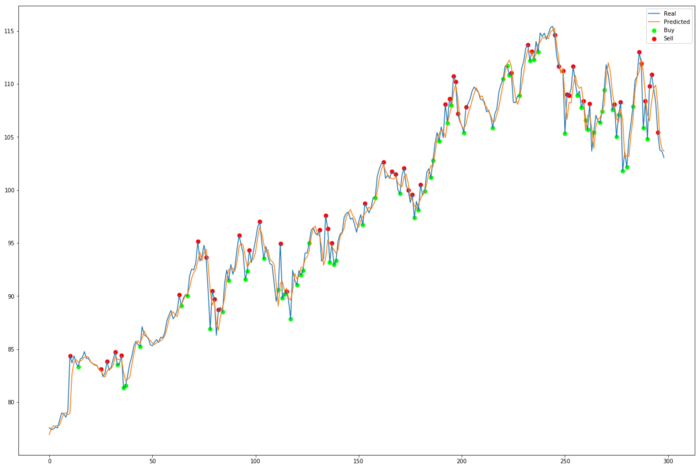
Member-only

·

Listen

# **Predicting stock prices using deep learning**

## If a human investor can be successful, why can’t a machine?

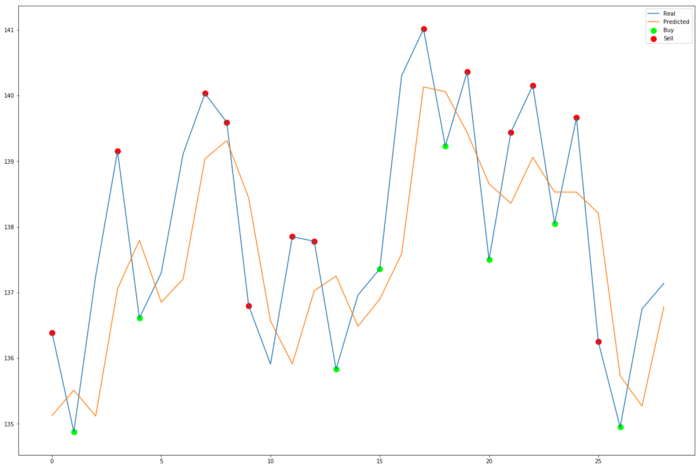


I would just like to add a disclaimer — this project is entirely intended for research purposes! I’m just a student learning about deep learning, and the project is a work in progress, please don’t put any money into an algorithm as naïve as this!

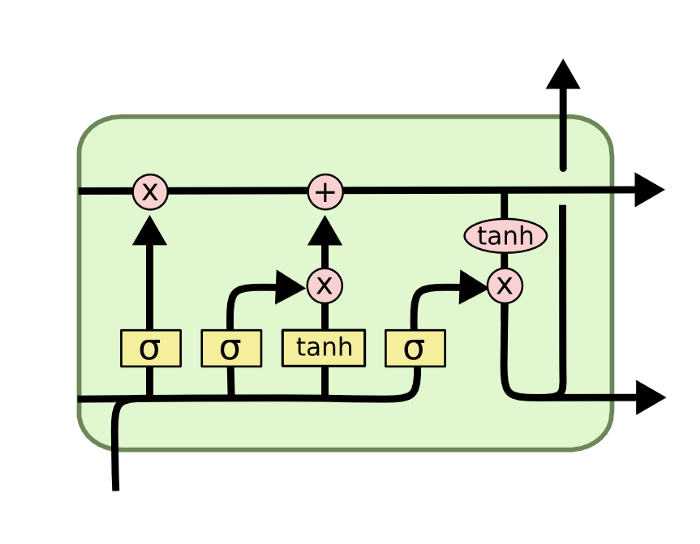
Algorithmic trading has revolutionised the stock market and its surrounding industry. Over 70% of all trades happening in the US right now are being handled by bots[1]. Gone are the days of the packed stock exchange with suited people waving sheets of paper [shouting into telephones](https://youtu.be/reFOV3r9T4s?t=107).

This got me thinking of how I could develop my own algorithm for trading stocks, or at least try to accurately predict them.

And the results weren’t bad!



I’ve learned a lot about neural networks and machine learning over the summer and one of the most recent and applicable ML technologies I learnt about is the LSTM cell [2].



<https://colah.github.io/posts/2015-08-Understanding-LSTMs/>

 This is achieved through the use of a recurrent node inside the LSTM cell. This node has an edge looping back on itself with a weight of one, meaning at every feedfoward iteration the cell can hold onto information from the previous step, as well as all previous steps. Since the looping connection’s weight is one, old memories wont fade over time like they would in traditional RNNs.

LTSMs and recurrent neural networks are as a result good at working with time series data thanks to their ability to remember the past. By storing some of the old state in these recurrent nodes, RNNs and LSTMs can reason about current information as well as information the network had seen one, ten or a thousand steps ago. Even better, I don’t have to write my own implementation of an LSTM cell; [they’re a default layer in Tensorflow’s Keras](https://keras.io/layers/recurrent/" \l "lstm).

So I had my plan; to use LSTMs and Keras to predict the stock market, and perhaps even make some money.

If you want to jump straight into the code you can check out the [GitHub repo](https://github.com/yacoubb/stock-trading-ml) :)

***Note from Towards Data Science’s editors:*** *While we allow independent authors to publish articles in accordance with our*[*rules and guidelines*](https://towardsdatascience.com/questions-96667b06af5)*, we do not endorse each author’s contribution. You should not rely on an author’s works without seeking professional advice. See our*[*Reader Terms*](https://towardsdatascience.com/readers-terms-b5d780a700a4) *for details.*

# **The Dataset**

The good thing about stock price history is that it’s basically a well labelled pre formed dataset. After some googling I found a service called [AlphaVantage](http://www.alphavantage.co/). They offered the daily price history of NASDAQ stocks for the past 20 years. This included the open, high, low, close and volume of trades for each day, from today all the way back up to 1999. Even better, a [python wrapper](https://github.com/RomelTorres/alpha_vantage) exists for the service. I got my free API key from the website and downloaded Microsofts daily stock history.

Since AlphaVantage’s free API only allows for 5 calls per minute (and up to 500 calls a day) I opted to download the datasets and save them in CSV format so I could use them as often as I wanted.

For the stocks that had their IPO listing within the past 20 years, the first day of trading that stock often looked anomalous due to the massively high volume. This inflated max volume value also affected how other volume values in the dataset were scaled when normalising the data, so I opted to drop the oldest data points out of every set. I also drop the date since the model doesn’t need to know anything about *when* trades happened, all it needs is well ordered time series data.

I also keep track of the number of {*history\_points*} we want to use; the number of days of stock history the model gets to base its predictions off of. So if history\_points is set to 50, the model will train on and require the past 50 days of stock history to make a prediction about just the next day.

Now we have to normalise the data — scale it between 0 and 1 — to improve how quickly our network converges[3] . Sklearn has a great *preprocessing* library capable of doing this.

*data\_normalised* now contains the normalised stock prices.

Now to get the dataset ready for model consumption.

The *ohlcv\_histories* list will be our x parameter when training the neural network. Each value in the list is a numpy array containing 50 open, high, low, close, volume values, going from oldest to newest. This is controlled by the *history\_points* parameter, as seen inside the slice operation.

So for each x value we are getting the [i : i + history\_points] stock prices (remember that numpy slicing is [inclusive:exclusive]), then the y value must be the singular [i + history\_points] stock price; the stock price for the very next day.

Here we also have to choose what value we are intending on predicting. I decided to predict the **open** value for the next day, so we need to get the 0-th element of every ohlcv value in the data, hence *data\_scaled[:,0]*.

There’s also a variable called y\_normaliser to hold on to. This is used at the end of a prediction, where the model will spit out a *normalised* number between 0 and 1, we want to apply the reverse of the dataset normalisation to scale it back up to real world values. In the future we will also use this to compute the real world (un-normalised) error of our model.

Then to get the data working with Keras I make the y array 2-dimensional by way of *np.expand\_dims()*. And finally I keep hold of the unscaled next day open values for plotting results later.

Just before returning the data we check that the number of x’s == the number of y’s.

Now we can get any dataset we have the csv file for by running:

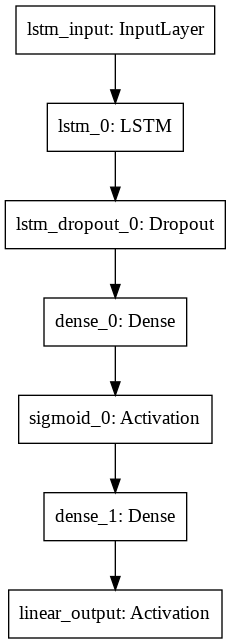
And that’s it! Our dataset is ready.

# **The Model**

I started this project only knowing how to write sequential Keras code, but I ended up learning it’s functional API since I wanted a more complex network structure, eventually featuring two inputs with different layer types in each branch.

I’ll go over the most basic model that I came up with first.

Which gives us a model that looks like:



The input layer has shape (history\_points, 5), since each input data point is an array shaped like [*history\_points***×**OHLCV]. The model has 50 LSTM cells in the first layer, a dropout layer to prevent overfitting and then some dense layers to bring all of the LSTM data together.

An important feature of this network is the linear output activation, allowing the model to tune its penultimate weights accurately.

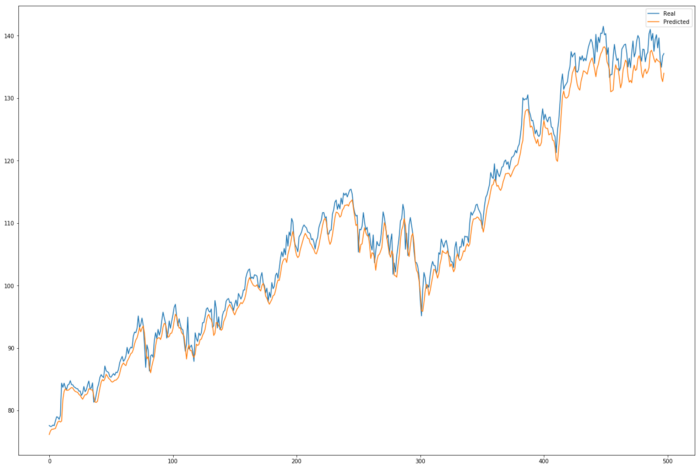
# **The Training**

This is why I love Keras. Literally 3 lines of code and you instantly know how well your model is doing on a dataset. I got a final evaluation score of 0.00029, which seems super low but remember that this is the mean squared error of the *normalised* data. After scaling this value will go up significantly, so it’s not a great metric for loss.

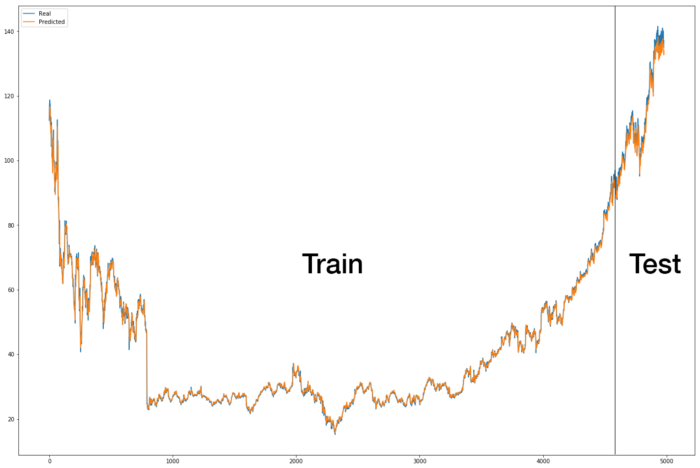
# **The Evaluation**

To more accurately evaluate the model, let’s see how it predicts the test set in comparison with the real values. First we scale the predicted values up, then we compute the mean squared error, but then to make the error relative to the dataset we divide it by the ‘spread’ of the test data — the max test data point minus the min.

This gives us an adjusted mean squared error of **7.17**. Is that good? It’s not amazing, it means on average the predicted line deviates over 7% from the actual. Let’s see how it looks on a graph.



But not bad! I’m not sure why the predicted value is consistently lower than the actual value, maybe it’s something to do with the way the test and train sets are split. And the results across the full dataset:



It’s hard to tell how well the algorithm is performing across this whole graph but you can definitely see a tighter fit across the train set, as we would expect.

# **The Improvements**

We could try to make our model more complex, and also increase the size of the dataset. Let’s start with trying to create a more complex model.

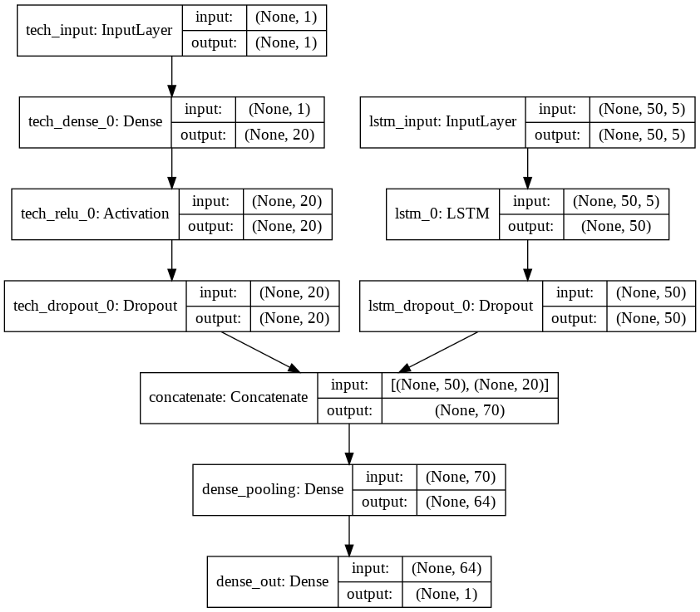
A common metric used by stock market analysts are *technical indicators[4]*. Technical indicators are math operations done on stock price history, and are traditionally used as visual aids to help identify the direction the market is going to change in. We can augment our model to accept these technical indicators through a secondary input branch.

For now let’s use only the simple moving average SMA indicator as an extra input into our network.

To calculate the simple moving average of a stock, simply take the average of the stock’s *closing* price over the past n time steps[5]. This works great for us, since we are already dealing with fixed timestep blocks of price history. To include the SMA in our model we will have to change our dataset handling code.

This happens just after we have defined the ohlcv\_histories and next\_day\_open\_values arrays. We loop across every 50-price block of data and calculate the mean of the 3rd column, the closing price, and add that value to a *technical\_indicators* list. The list then goes through the same transformations as the rest of the data, being scaled to fit within the values 0 to 1. We then change the return statement to return the technical indicators, as well as the other stuff we returned from before.

Now to augment the model to match this new dataset. We want to be able to use our old LSTM structure, but incorporate the SMA technical indicator somewhere in the mix. Since the SMA is not a time-series piece of data we shouldn’t pass it through the LSTM. Instead we should mix it in before the final prediction is made; we should input it into the penultimate 64-node dense layer. So we will need a model with two inputs, a concatenation layer and one output.

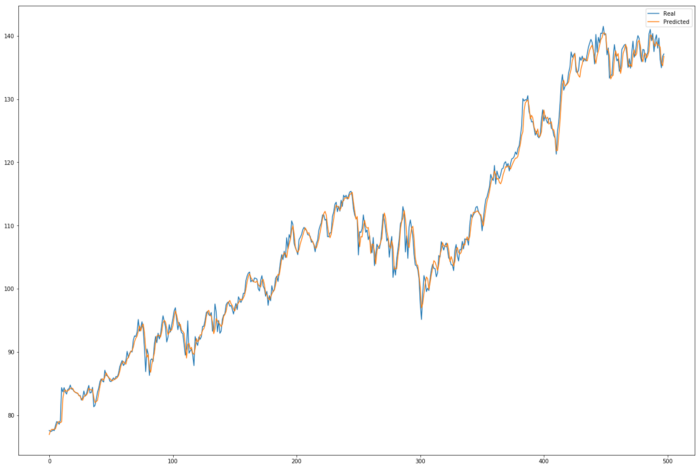


Note how we used technical\_indicators.shape[1] as the input shape for the tech\_input layer. This means that any new technical indicators we add will fit in just fine when we recompile the model.

The evaluation code has to be changed to match this dataset change as well.

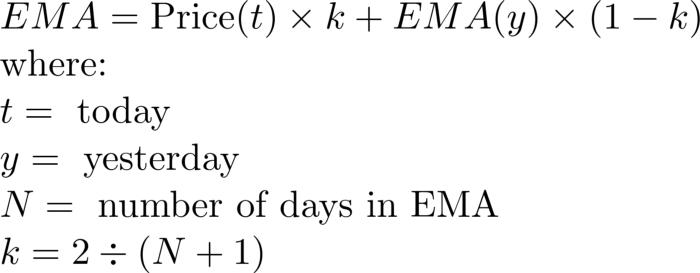
We pass in a list of [ohlcv, technical\_indicators] as the input to our model. This order matches the way we defined our model’s input.

And we get an adjusted mean squared error of **2.25**! Much lower, and the prediction appears to fit significantly closer to the test set when plotted;



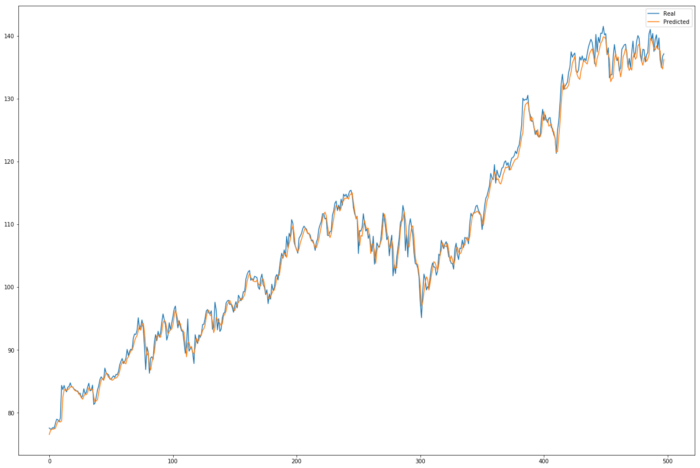
This model appears to not suffer the previous problem of being continuously off by a fixed amount, but does seem to suffer from not catching sudden jumps as well. Like at around x-coordinate 120, a large jump and dip in the real price occurs but the model fails to capture this effectively. But it is getting better! And it seems that technical indicators could be the way forward.

Let’s try including a more advanced technical indicator: the Moving Average Convergence Divergence. The MACD is calculated by subtracting the 26-period Exponential Moving Average from the 12-period EMA[6]. The EMA is calculated[7] using the formula:



To update our technical indicators loop to include the MACD indicator:

With the SMA and MACD indicators, our model achieves an adjusted mean squared error of 2.60. Slightly higher than when using just the SMA and that is reflected in the graph.



You can see that it doesn’t fit as accurately as when using just the SMA. Maybe since we’re giving the model more data it needs more layers to make sense of it, but for now I’m just going to omit the use of the MACD technical indicator.

We can also experiment with using a larger dataset. If we assume that the techniques applied to stock prediction for Microsoft’s stock can be generalised to all stocks, then we could just combine the results of the csv\_to\_dataset() function for lots of different stock histories. As an example, we could train on the stock histories of AMZN, FB, GOOGL, MSFT, NFLX, and test the results on the AAPL stock.

First run the save\_dataset method from all the way back at the start of this post for all of the different stocks you wish to use. Then, the dataset creation method would look like:

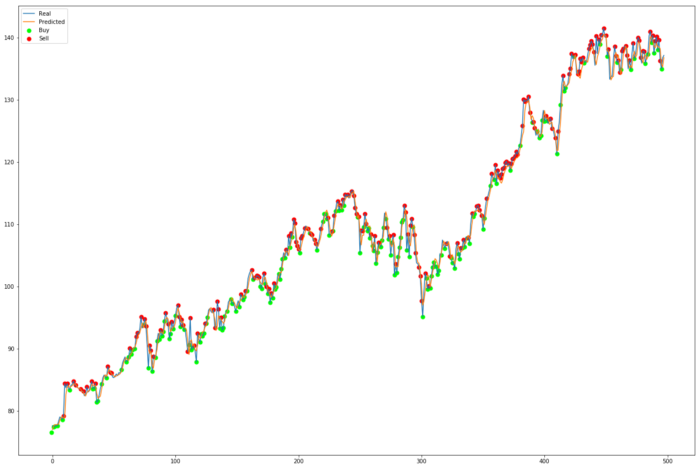
This basically says for each csv file in the current directory, if the file name is not *test\_set\_name*, load the dataset from it and append it to the whole dataset. Then load the *test\_set\_name* csv file and use that as the test dataset. Using the AMZN, NFLX, GOOGL, FB and MSFT stock prices for the train set we get 19854 train samples. Using the AAPL stock for the test set we get 4981 test samples.

After training with this new, larger dataset for 50 epochs with the SMA indicator we get an adjusted MSE value of **12.97**, higher than when we trained on just one stock. Maybe it’s the case that stock prediction is actually symbol-specific; that different stocks may behave in slightly different ways. It’s clear that they all fundamentally behave similarly, since the network was able to learn and generalise from different stocks, but for the most accurate predictions it might be best to only train on the history of that stock. After all, the Microsoft dataset seemed to have enough training data to figure it out.

# **The Algorithm**

Armed with an okay-ish stock prediction algorithm I thought of a naïve way of creating a bot to decide to buy/sell a stock today given the stock’s history. In essence you just predict the opening value of the stock for the next day, and if it is beyond a threshold amount you buy the stock. If it is below another threshold amount, sell the stock. This dead simple algorithm actually seemed to work quite well — visually at least.

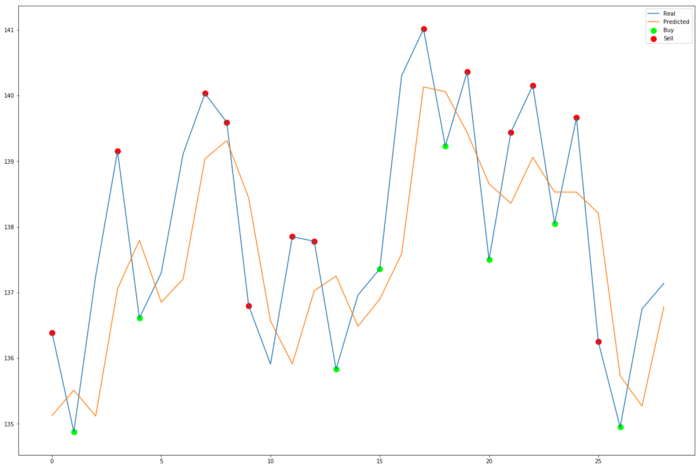
Then plot the trades.



Looking good! The algorithm appears to be correctly buying low and selling high. Remember, this is all on test data — data that the network has never seen before. There’s no reason why this data couldn’t have been live, and these trades actually be real!

Given these buys and sells, if we say that at each ‘buy’ we buy up $10 worth of the stock, and at each ‘sell’ we sell all of the stock, the algorithm would have earned $38.47. But bear in mind that is across 500 days. The code to calculate the earnings of the algorithm is here;

If we instead tried this algorithm over 30 days, using the same $10 purchasing amount and a threshold level of 0.2, the algorithm would have earned a measly $1.55. But that’s better than nothing!



# **The Conclusion**

I think there is still some room for improvement for the prediction algorithm. Namely, the technical indicators used, history\_points hyperparameter, buy/sell algorithm/hyperparameters and model architecture are all things that I would like to optimise in the future.

I’d also like to look into giving the model more data by having more LSTM branches, one for each timestep available on AlphaVantage, so the network can make decisions based on short, medium and long term trends.

The full code for this project is available on my [GitHub](https://github.com/yacoubb/stock-trading-ml). Feel free to leave any feedback/improvements over on the issues page!

I do plan to expand on this project some more, to really push the limits of what can be achieved using just numerical data to predict stocks. Keep up to date with what I’m doing on my [blog](https://yacoubahmed.me/blog)!

# **References**

[1]: <https://www.experfy.com/blog/the-future-of-algorithmic-trading>

[2]: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>

[3]: <https://jovianlin.io/why-is-normalization-important-in-neural-networks/>

[4]: <https://www.investopedia.com/terms/t/technicalindicator.asp>

[5]: <https://www.investopedia.com/terms/s/sma.asp>

[6]: <https://www.investopedia.com/terms/m/macd.asp>

[7]: <https://www.investopedia.com/ask/answers/122314/what-exponential-moving-average-ema-formula-and-how-ema-calculated.asp>

24

24

## **Sign up for The Variable**

### By Towards Data Science

Every Thursday, the Variable delivers the very best of Towards Data Science: from hands-on tutorials and cutting-edge research to original features you don't want to miss. [Take a look.](https://medium.com/towards-data-science/newsletters/the-variable?source=newsletter_v3_promo--------------------------newsletter_v3_promo-----------)

By signing up, you will create a Medium account if you don’t already have one. Review our [Privacy Policy](https://policy.medium.com/medium-privacy-policy-f03bf92035c9?source=newsletter_v3_promo--------------------------newsletter_v3_promo----------d6fe9076899-) for more information about our privacy practices.

[Get this newsletter](https://medium.com/m/signin?actionUrl=%2F_%2Fapi%2Fsubscriptions%2Fnewsletters%2Fd6fe9076899&operation=register&redirect=https%3A%2F%2Fmedium.com%2Ftowards-data-science%2Fnewsletters%2Fthe-variable&collection=Towards+Data+Science&collectionId=7f60cf5620c9&newsletterV3=The+Variable&newsletterV3Id=d6fe9076899&user=Ludovic+Benistant&userId=895063a310f4&source=--------------------------newsletter_v3_promo----------d6fe9076899-)

## [More from Towards Data Science](https://towardsdatascience.com/?source=post_page-----696802da94fe--------------------------------)

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fcollection%2Ftowards-data-science%2F696802da94fe&operation=register&redirect=https%3A%2F%2Ftowardsdatascience.com%2Fgetting-rich-quick-with-machine-learning-and-stock-market-predictions-696802da94fe&collection=Towards+Data+Science&collectionId=7f60cf5620c9&source=post_page-----696802da94fe---------------------follow_footer-----------)

Your home for data science. A Medium publication sharing concepts, ideas and codes.

[](https://robertosalazarr.medium.com/?source=post_page-----696802da94fe----0----------------------------)

[Roberto Salazar](https://robertosalazarr.medium.com/?source=post_page-----696802da94fe----0----------------------------)

[·Oct 11, 2019](https://towardsdatascience.com/r-for-industrial-engineers-quality-control-charts-p-chart-and-np-chart-e99b2f07fbe1?source=post_page-----696802da94fe----0----------------------------)

Member-only

## [Quality Control Charts: p-chart and np-chart](https://towardsdatascience.com/r-for-industrial-engineers-quality-control-charts-p-chart-and-np-chart-e99b2f07fbe1?source=post_page-----696802da94fe----0----------------------------)

[Creating Quality Control Charts using “qcc” R package — Quality Control Charts Quality control charts represent a great tool for engineers to monitor if a process is under statistical control. They help visualize variation, find and correct problems when they occur, predict expected ranges of outcomes and analyze patterns of process variation from special or common causes. …](https://towardsdatascience.com/r-for-industrial-engineers-quality-control-charts-p-chart-and-np-chart-e99b2f07fbe1?source=post_page-----696802da94fe----0----------------------------)

[Data Science](https://medium.com/tag/data-science?source=post_page-----696802da94fe---------------data_science-----------------)

[3 min read](https://towardsdatascience.com/r-for-industrial-engineers-quality-control-charts-p-chart-and-np-chart-e99b2f07fbe1?source=post_page-----696802da94fe----0----------------------------)

[](https://towardsdatascience.com/r-for-industrial-engineers-quality-control-charts-p-chart-and-np-chart-e99b2f07fbe1?source=post_page-----696802da94fe----0----------------------------)

Share your ideas with millions of readers.

[Write on Medium](https://medium.com/new-story?source=post_page_footer_cta_write-------------------------------------)

[](https://medium.com/@lovelytics?source=post_page-----696802da94fe----1----------------------------)

[Lovelytics](https://medium.com/@lovelytics?source=post_page-----696802da94fe----1----------------------------)

[·Oct 11, 2019](https://towardsdatascience.com/a-guide-to-integrating-text-analytics-into-tableau-94d1a1331117?source=post_page-----696802da94fe----1----------------------------)

Member-only

## [A Guide to Integrating Text Analytics into Tableau](https://towardsdatascience.com/a-guide-to-integrating-text-analytics-into-tableau-94d1a1331117?source=post_page-----696802da94fe----1----------------------------)

[Data is often dirty and messy. Sometimes, it doesn’t even come in the right form for quick analysis and visualization. While Tableau (and Prep) had several tools to deal with numeric, categorical, and even spatial data, one consistent missing piece was handling unstructured text data. Not anymore. In the latest…](https://towardsdatascience.com/a-guide-to-integrating-text-analytics-into-tableau-94d1a1331117?source=post_page-----696802da94fe----1----------------------------)

[Data Science](https://medium.com/tag/data-science?source=post_page-----696802da94fe---------------data_science-----------------)

[12 min read](https://towardsdatascience.com/a-guide-to-integrating-text-analytics-into-tableau-94d1a1331117?source=post_page-----696802da94fe----1----------------------------)

[](https://towardsdatascience.com/a-guide-to-integrating-text-analytics-into-tableau-94d1a1331117?source=post_page-----696802da94fe----1----------------------------)

[](https://medium.com/@tylerwmarrs?source=post_page-----696802da94fe----2----------------------------)

[Tyler Marrs](https://medium.com/@tylerwmarrs?source=post_page-----696802da94fe----2----------------------------)

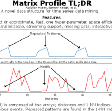
[·Oct 11, 2019](https://towardsdatascience.com/introduction-to-matrix-profiles-5568f3375d90?source=post_page-----696802da94fe----2----------------------------)

## [Introduction to Matrix Profiles](https://towardsdatascience.com/introduction-to-matrix-profiles-5568f3375d90?source=post_page-----696802da94fe----2----------------------------)

[A Novel Data Structure for Mining Time Series — In time series analysis, one is typically interested in two things; anomalies and trends. For example, a physician examines an EKG (electrocardiogram — heartbeat reading) for anomalous events that indicate at-risk patients. An individual working within retail needs to understand what items sell and when they sell (seasonality) to increase…](https://towardsdatascience.com/introduction-to-matrix-profiles-5568f3375d90?source=post_page-----696802da94fe----2----------------------------)

[Data Science](https://medium.com/tag/data-science?source=post_page-----696802da94fe---------------data_science-----------------)

[7 min read](https://towardsdatascience.com/introduction-to-matrix-profiles-5568f3375d90?source=post_page-----696802da94fe----2----------------------------)

[](https://towardsdatascience.com/introduction-to-matrix-profiles-5568f3375d90?source=post_page-----696802da94fe----2----------------------------)

[](https://medium.com/@johngatopoulos?source=post_page-----696802da94fe----3----------------------------)

[Ioannis Gatopoulos](https://medium.com/@johngatopoulos?source=post_page-----696802da94fe----3----------------------------)

[·Oct 11, 2019](https://towardsdatascience.com/line-detection-make-an-autonomous-car-see-road-lines-e3ed984952c?source=post_page-----696802da94fe----3----------------------------)

Member-only

## [Line Detection: Make an Autonomous Car see Road Lines](https://towardsdatascience.com/line-detection-make-an-autonomous-car-see-road-lines-e3ed984952c?source=post_page-----696802da94fe----3----------------------------)

[Step by step you can turn a video stream into a line detector via Computer Vision Motivation Fully self-driving passenger cars are not “just around the corner”. Elon Musk claims that Teslas will have a “full self-driving” capability by the end of 2020. Especially, he says that Tesla’s hardware is already…](https://towardsdatascience.com/line-detection-make-an-autonomous-car-see-road-lines-e3ed984952c?source=post_page-----696802da94fe----3----------------------------)

[Computer Vision](https://medium.com/tag/computer-vision?source=post_page-----696802da94fe---------------computer_vision-----------------)

[6 min read](https://towardsdatascience.com/line-detection-make-an-autonomous-car-see-road-lines-e3ed984952c?source=post_page-----696802da94fe----3----------------------------)

[](https://towardsdatascience.com/line-detection-make-an-autonomous-car-see-road-lines-e3ed984952c?source=post_page-----696802da94fe----3----------------------------)

[](https://medium.com/@kanellisg?source=post_page-----696802da94fe----4----------------------------)

[Kanellis Georgios](https://medium.com/@kanellisg?source=post_page-----696802da94fe----4----------------------------)

[·Oct 11, 2019](https://towardsdatascience.com/proper-balancing-for-cross-validation-d95c17ff0ab4?source=post_page-----696802da94fe----4----------------------------)

Member-only

## [Proper Balancing for Cross Validation](https://towardsdatascience.com/proper-balancing-for-cross-validation-d95c17ff0ab4?source=post_page-----696802da94fe----4----------------------------)

[Who hasn’t come across the need of applying the cross validation technique, while the dataset in hand is imbalanced, in regards to the number of instances per target class value. The question here is, do we apply it properly? The purpose of this article is to show a way to…](https://towardsdatascience.com/proper-balancing-for-cross-validation-d95c17ff0ab4?source=post_page-----696802da94fe----4----------------------------)

[Machine Learning](https://medium.com/tag/machine-learning?source=post_page-----696802da94fe---------------machine_learning-----------------)

[7 min read](https://towardsdatascience.com/proper-balancing-for-cross-validation-d95c17ff0ab4?source=post_page-----696802da94fe----4----------------------------)

[](https://towardsdatascience.com/proper-balancing-for-cross-validation-d95c17ff0ab4?source=post_page-----696802da94fe----4----------------------------)

[Read more from Towards Data Science](https://towardsdatascience.com/?source=post_page-----696802da94fe--------------------------------)

## Recommended from Medium

[](https://ramji-b.medium.com/?source=post_internal_links---------0----------------------------)

[Ramji Balasubramanian](https://ramji-b.medium.com/?source=post_internal_links---------0----------------------------)

in

[Analytics Vidhya](https://medium.com/analytics-vidhya?source=post_internal_links---------0----------------------------)

## [Signature Recognition Using OpenCV -Part 2](https://medium.com/analytics-vidhya/signature-recognition-using-opencv-part-2-62685586d566?source=post_internal_links---------0----------------------------)

[](https://medium.com/analytics-vidhya/signature-recognition-using-opencv-part-2-62685586d566?source=post_internal_links---------0----------------------------)

[](https://wilsonlouie.medium.com/?source=post_internal_links---------1----------------------------)

[Wilson Louie](https://wilsonlouie.medium.com/?source=post_internal_links---------1----------------------------)

in

[Artificial Intelligence in Plain English](https://ai.plainenglish.io/?source=post_internal_links---------1----------------------------)

## [TensorFlow Model Serving](https://ai.plainenglish.io/machine-learning-on-akash-decloud-part-2-3-tensorflow-model-serving-12e30d77a156?source=post_internal_links---------1----------------------------)

[](https://ai.plainenglish.io/machine-learning-on-akash-decloud-part-2-3-tensorflow-model-serving-12e30d77a156?source=post_internal_links---------1----------------------------)

[](https://medium.com/@mobrien1080?source=post_internal_links---------2----------------------------)

[Mike O'Brien](https://medium.com/@mobrien1080?source=post_internal_links---------2----------------------------)

in

[Apache MXNet](https://medium.com/apache-mxnet?source=post_internal_links---------2----------------------------)

## [How Cimpress Delivers Cloud Inference for its Image Processing Services](https://medium.com/apache-mxnet/how-cimpress-delivers-cloud-inference-for-its-image-processing-services-c38951732f97?source=post_internal_links---------2----------------------------)

[](https://medium.com/apache-mxnet/how-cimpress-delivers-cloud-inference-for-its-image-processing-services-c38951732f97?source=post_internal_links---------2----------------------------)

[](https://medium.com/@impiyush?source=post_internal_links---------3----------------------------)

[Piyush Agarwal](https://medium.com/@impiyush?source=post_internal_links---------3----------------------------)

in

[Towards Data Science](https://towardsdatascience.com/?source=post_internal_links---------3----------------------------)

## [End-to-End Machine Learning Project: Part-1](https://towardsdatascience.com/end-to-end-machine-learning-project-part-1-806d04da8deb?source=post_internal_links---------3----------------------------)

[](https://towardsdatascience.com/end-to-end-machine-learning-project-part-1-806d04da8deb?source=post_internal_links---------3----------------------------)

[](https://medium.com/@jashr2312?source=post_internal_links---------4----------------------------)

[Jash Rathod](https://medium.com/@jashr2312?source=post_internal_links---------4----------------------------)

in

[Analytics Vidhya](https://medium.com/analytics-vidhya?source=post_internal_links---------4----------------------------)

## [How to make self-learning games using Reinforcement Learning.](https://medium.com/analytics-vidhya/how-to-make-self-learning-games-using-reinforcement-learning-dee2f8904b71?source=post_internal_links---------4----------------------------)

[](https://medium.com/analytics-vidhya/how-to-make-self-learning-games-using-reinforcement-learning-dee2f8904b71?source=post_internal_links---------4----------------------------)

[](https://prasantdixit.medium.com/?source=post_internal_links---------5----------------------------)

[Prasant Kumar](https://prasantdixit.medium.com/?source=post_internal_links---------5----------------------------)

in

[Artificialis](https://medium.com/artificialis?source=post_internal_links---------5----------------------------)

## [Parameters in Convolutional Network](https://medium.com/artificialis/parameters-in-convolutional-network-828cbe7f0c78?source=post_internal_links---------5----------------------------)

[](https://medium.com/artificialis/parameters-in-convolutional-network-828cbe7f0c78?source=post_internal_links---------5----------------------------)

[](https://medium.com/@adel_dajani?source=post_internal_links---------6----------------------------)

[T.J.Adel Dajani](https://medium.com/@adel_dajani?source=post_internal_links---------6----------------------------)

## [Can a computer be smart enough to Identify car sides?](https://medium.com/@adel_dajani/can-a-computer-be-smart-enough-to-identify-car-sides-e73b6ec5ab99?source=post_internal_links---------6----------------------------)

[](https://medium.com/@adel_dajani/can-a-computer-be-smart-enough-to-identify-car-sides-e73b6ec5ab99?source=post_internal_links---------6----------------------------)

[](https://medium.com/@atulbalaji318?source=post_internal_links---------7----------------------------)

[Atul Balaji](https://medium.com/@atulbalaji318?source=post_internal_links---------7----------------------------)

## [Anchor based and Keypoint based Object Detection — by Amrit Parimi & Atul Balaji](https://medium.com/@atulbalaji318/anchor-based-and-keypoint-based-object-detection-by-amrit-parimi-atul-balaji-81b7486a2a13?source=post_internal_links---------7----------------------------)

[](https://medium.com/@atulbalaji318/anchor-based-and-keypoint-based-object-detection-by-amrit-parimi-atul-balaji-81b7486a2a13?source=post_internal_links---------7----------------------------)

[About](https://medium.com/about?autoplay=1&source=post_page-----696802da94fe--------------------------------)[Help](https://help.medium.com/hc/en-us?source=post_page-----696802da94fe--------------------------------)[Terms](https://policy.medium.com/medium-terms-of-service-9db0094a1e0f?source=post_page-----696802da94fe--------------------------------)[Privacy](https://policy.medium.com/medium-privacy-policy-f03bf92035c9?source=post_page-----696802da94fe--------------------------------)