Machine Learning EngineerNano degree Capstone Project

Predicting Bitcoin Price with LSTM neural network.

1) Definition

Project Overview

I was always interested in algorithmic trading and wanted to write my own automated trading strategy. However, I think that stock/bond and FX markets are oversaturated with institutional traders with far greater access to better technology, much larger trading capital and better equipped talent pool for me to stay competitive. The cryptocurrency space which has experienced a significant growth in the last few years with over 2000+ new coins being created since 2013 are considered too volatile and risky for institutional traders might be a perfect opportunity to exploit the trading opportunities that are no longer available in more conventional markets. Elendner et al. (2017) proved that cryptocurrencies are uncorrelated with each other and with traditional assets.

Problem Statement

The purpose of this project is to create a deep learning model that is able to predict the Price of the Cryptocurrency Coin with the use of historical price data available online. The idea is to build a coin price predictor which would be able to use historical market data such as Opening price, Daily Highs and Lows and Trading volume over some period of time to predict the closing price of a coin at the end of the day. The output of the model would be used to make a trading decision for buying or selling of one unit of coin for one day. I will assume that there is no market spread and no trading fees associated with the transaction and that short selling (betting that the market will fall) is allowed with no additional fees.

Metrics

This project can be divided into two distinct parts:

1) Bitcoin price prediction with an LSTM model is a regression problem, so the mean squared error would be the metric of choice.

The room of mean squared error gives the average of the deviation of the predicted value from the true value, which can be compared with the mean of the true value and measure the deviation from the truth.

2) I will deal with the trading strategy as with a classification problem. Since its really hard to predict the price of financial assets I will be only using the sign from the model prediction to make the purchase decision i.e. is the sign is positive I will buy one unit of bitcoin an if the sign is negative I will sell one unit respectively. The difference between the actual closing price of today and the closing price of the previous day would be considered as profit/loss for the day.
I will compare the results of my trading strategy with the simple buy and hold strategy i.e buy one unit at the beginning and hold it until the and.

2) Analysis

Data Exploration

I am using the daily price data of Bitcoin, which I have obtained from https://coinmarketcap.com. The Dataset includes the following variables:

Date: the trading date
Open: the opening price
High: the highest price for the da
Low: the lowest price for the day
Close: the latest price in the date range
Volume: number of coins trader in that day
Market Cap: number of coins * value

There are a total of 2369 daily observations starting from 2013-04-28 to 2019-10-22. All of the columns except for Date should be numeric, however they are all represented as string values. The data does not contain missing values, however some may appear when we will try to convert all of the columns (except for Date) to type: float. At that point I will impute them with the median values for that column beca neural networks cannot take missing values as input.

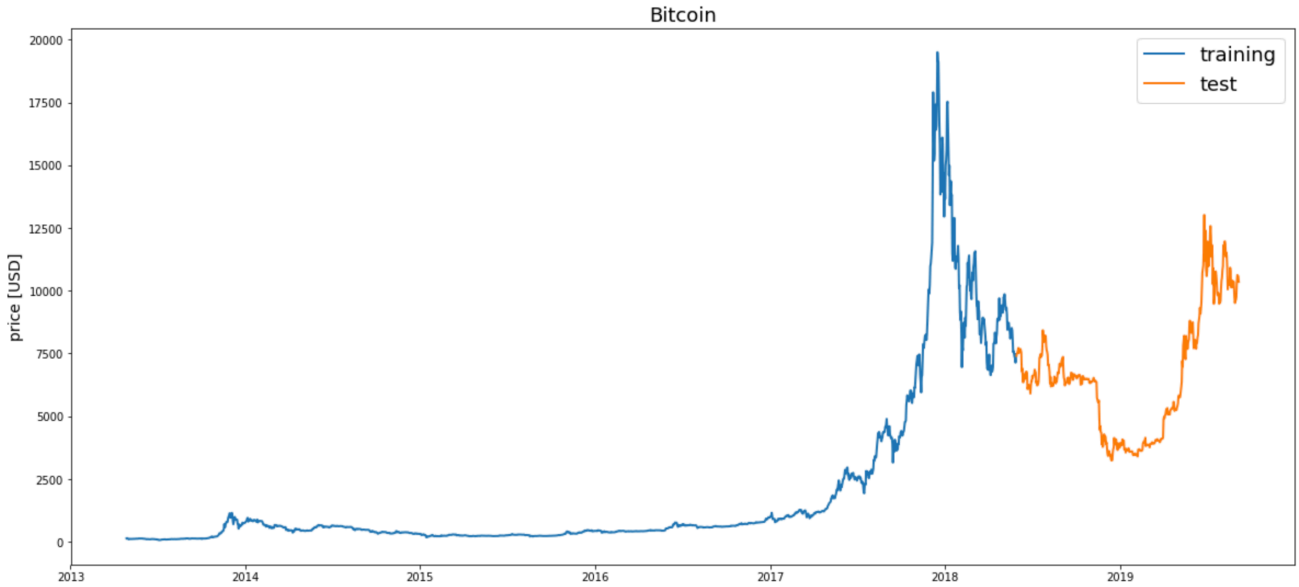
	Date	Open*	High	Low	Close**	Volume	Market Cap
0	Oct 22, 2019	8243.40	8296.65	8074.46	8078.20	16803377857	145469660005
1	Oct 21, 2019	8225.12	8296.69	8196.42	8243.72	15868748866	148432733155
2	Oct 20, 2019	7997.81	8281.82	7949.44	8222.08	15504249442	148026597064
3	Oct 19, 2019	7973.80	8082.63	7944.78	7988.56	13797825640	143808163807
4	Oct 18, 2019	8100.93	8138.41	7902.16	7973.21	15651592610	143517630376

First 5 rows from the unprocessed market data:
Since we are dealing with prices of unregulated financial asset, we can expect to have a very volatile and noisy time series. In a classical time series analysis on would be expected to make the time series stationary by taking a natural log and calculating a price difference before fitting a model, but since we are going to use a neural network this step wont be necessary and we can just scale the data between 0 and 1.

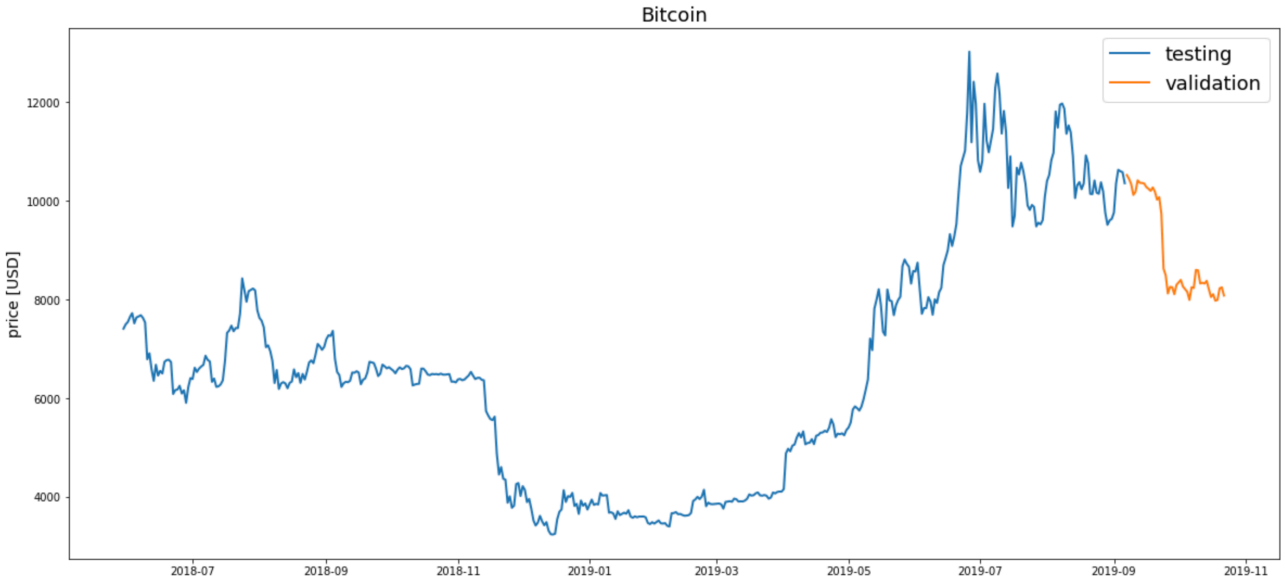
Data Visualisation

I am planning to split the data into training, testing and the validation datasets.

The training set will contain 1858 observations and will contain values from the beginning of the data 2013-04-28 to 2018-05-29 (Depicted by the blue line in the plot below). The tasing dataset would be the period straight after the train set, which will contain 465 observations and will go on until 2019-09-06 (depicted by the yellow line below)



The validation dataset on which we will test out our trading strategy will be at the very end of the time series and will contain 46 observations (yellow line below)



Algorithms and Techniques

I will use a Long Short Term Memory (LSTM) Neural network as a bitcoin price predictor in this project. I will use the Close price as the target variable and the rest of the valuables (except for date) as my independent variables. I will first scale the data between 0 and 1 with the “sklearn” MinMaxScaler then will convert the input into sequences. I will tune the sequence length from 1 to 10 and will use the one that gets the best result. The prediction will be 1 day ahead Close price of the bitcoin.

Benchmark

The performance of the model would be measured on how does the trading strategy based on the models predictions would perform against the simple buy and hold strategy on the validation dataset. The validation dataset starts at 2019-09-06, but since we ended up using the sequence length of 7 days, we can only start using out model on 2019-09-15, so this will also be the date will start using the wait and hold strategy.
The idea behind the wait and hold is simple. We are assuming to buy one unit of bitcoin at 2019-09-14 at the closing price and hold on to it until the end of the period, which would end on the 2019-10-22. The difference between the closing price from 2019-09-14 to 2019-10-22 would be considered as the P/L for the wait and hold strategy. This is the metric we will be trying to beat with the LSTM model.

3) Methodology

Data Preprocessing

First we have to obtain the market data. I using a website called https://coinmarketcap.com to get the daily bitcoin prices. I did not specify from/to dates so it will return the prices from the earliest date available until the latest date available, at the time of writing this happens to be from 2013-10-23 to 2019-10-22.

We need to clean the data. I am stating with the column names. They are all start with a capital letter and 'Open' and 'Close' columns have an Asterix next to them, so I removed the Asterix and converted all of the names to lower case.

The Date column is in a string format 'Oct 23, 2019', so I convert it to the 'yyyy-mm-dd' datetime.date() format.

All of the numeric columns are represented as strings in dataset. In order to do further transformation we will have to convert them all to double precision.

After that we check if we have any missing values in the dataset. There are several ways of dealing with missing data which ranges from dropping entire rows or imputing it with a statistic. Since we only have one missing value I decided to impute it with the median.

The next thing I want to do is reverse the data frame so the dates would from lowest to highest, since this is the order in which we are going to feed the data to the model. Since we are not going to use the date as a predictor variable I will turn it to an index so it will be easier to query.

Next step is to split the data into the trading, testing and the validation datasets. The train data would be fore the period from 2013 to 2018. The testing data would start at 2018-05-29 and go until 2019-09-06 mad the remains 46 days would be saved for the validation dataset.

The bitcoin price has changed a lot. It was as low as USD 100 in 2013 to as high as 20000 in 2018. This variation would make it hard for the neural network to converge. Do in order to make it easier we are going to Scale the data between 0 and 1. I will use the Sklearn.MinMaxScaler() and will app it to all of the numeric columns of the dataset. The procedure is as follows: we create the scaler object from the training dataset and then apply it to train testing and validation datasets. This would use the Min and Max values from the train data and will prevent the data leakage when scaling the testing and validation datasets.

Finally, we have to convert our data into sequences. I tried to train the data on sequence lengths from 1 to 10. The performance

on the testing set was more or less similar between all of them, is I went with the sequence Length of 7 for the final model, which would represent one trading week (Bitcoin trades on weekends).

Implementation

For this project I have train a LSTM Neural Network. I ended up using Keras library with Tensorflow backend to build my model. The Network takes a 3 dimensional array from the data preprocessing as input. The first dimension is the number of samples in the training batch, which is 4. The second dimension is the number of sequences in on train sample, which in our case is 7. The last dimension is the number of variables in the sample, which is 6. Then final network contains 3 LSTM layers with 50 neutrons in each one, each last layer is followed by a dropout layer with the with probably of dropout being 0.25. After that there is one fully connected layer with the output size of 1. I have used TanH as my activation function. I have used the Mean Squared Error as my Loss function, mainly because the data is volatile and MSE puts higher values to the high variance errors. Nesterov ADAM was used as my Optimiser, which is ADAM with the nesters Momentum.
I have trained the model for 50 epochs with shuffling the samples in the batches to prevent overfitting.
We can see the difference between the training and testing error on the graph below.

Refinement

I had to tune several Hyper parameters during the training process.

The first hyper parameter was the sequence length. I have tried all sequence lengths between the 1 and 10, but there was no

significant difference between them so I need up going with the sequence length of 7, which represents one trading week.

The second hyper parameter was the number of neutrons in the hidden layer. I have tried values of 50-100. The model started to overfit significantly on the 100 neutrons. Im assuming that this is because we only have 2000 training samples.

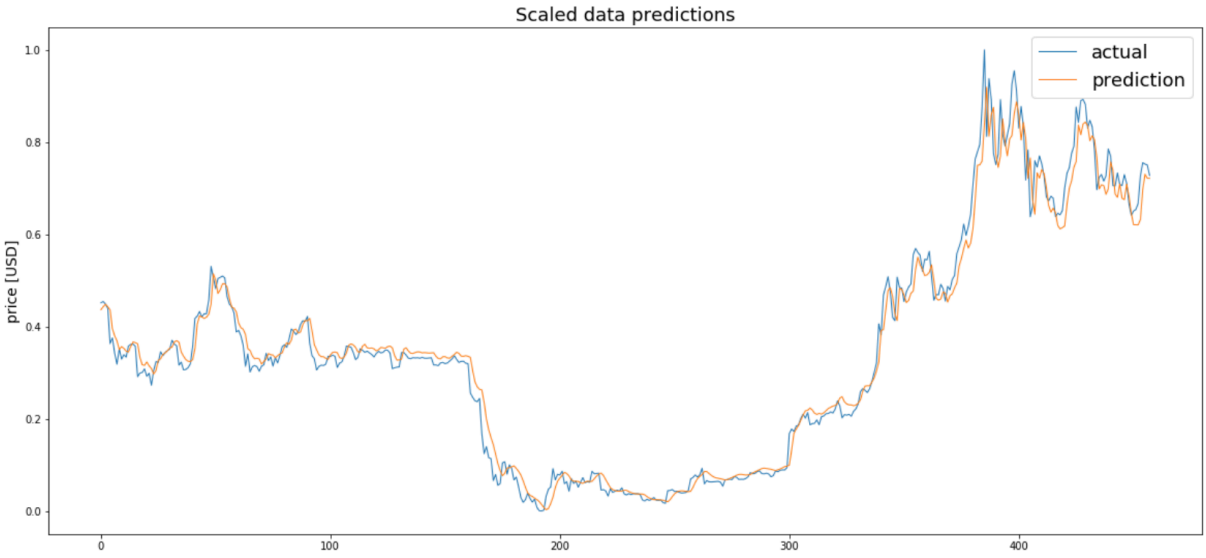
The third Hyper parameter was the number of training epochs. I have tried 20-50-100. The results were pretty much the same with the slight decrease of testing loss for the 50 epochs.

The fourth hyper parameter was the Optimiser. I tried both ADAM and Nesterov ADAM. The results were similar, but the is stronger support in the literature for the NADAM so I went with that.

4) Results

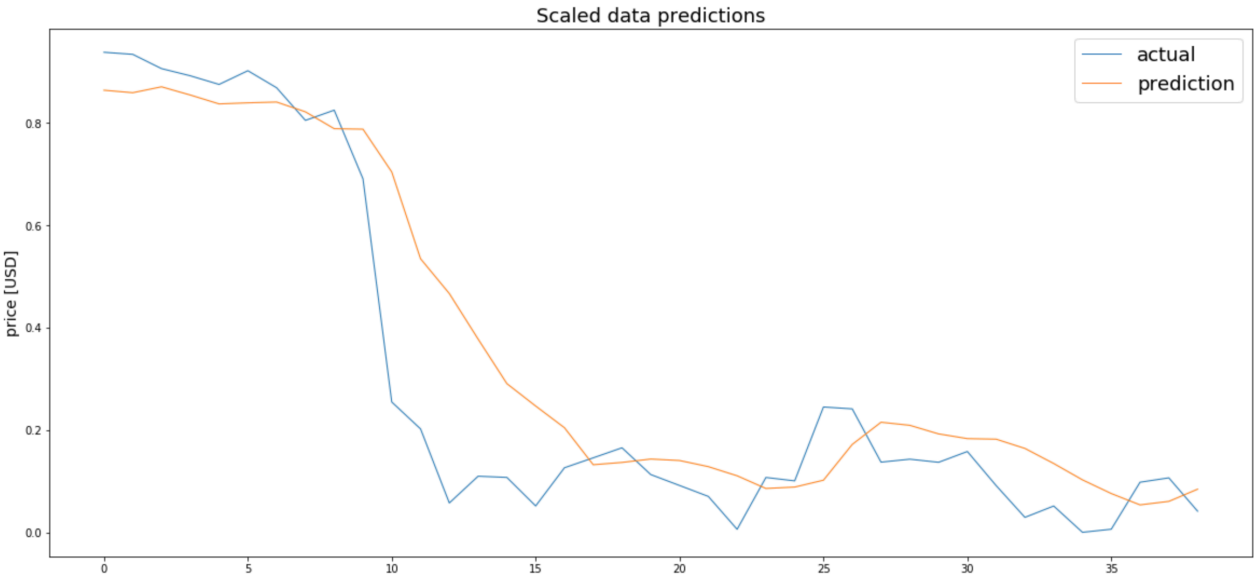
Model Evaluation and Validation

After building the model we can test it out on the testing and validation set.
The model performs very well on the **testing** set giving the Mean Absolute Error of 0.023065848984746132.
The graph below shows the actual bitcoin prices in blue colour and the predictions in yellow.



From the image above we can see that the model does predict better in the periods of small volatility and starts to performs worse when volatility is higher like during the summer of 2019.

The model performs slightly worse on the **validation** dataset yielding and AME of 0.09699825260978394, which makes sense since the model never seen this data before. From the image below we can see that the overall direction remain the same but the predictions are far less accurate.



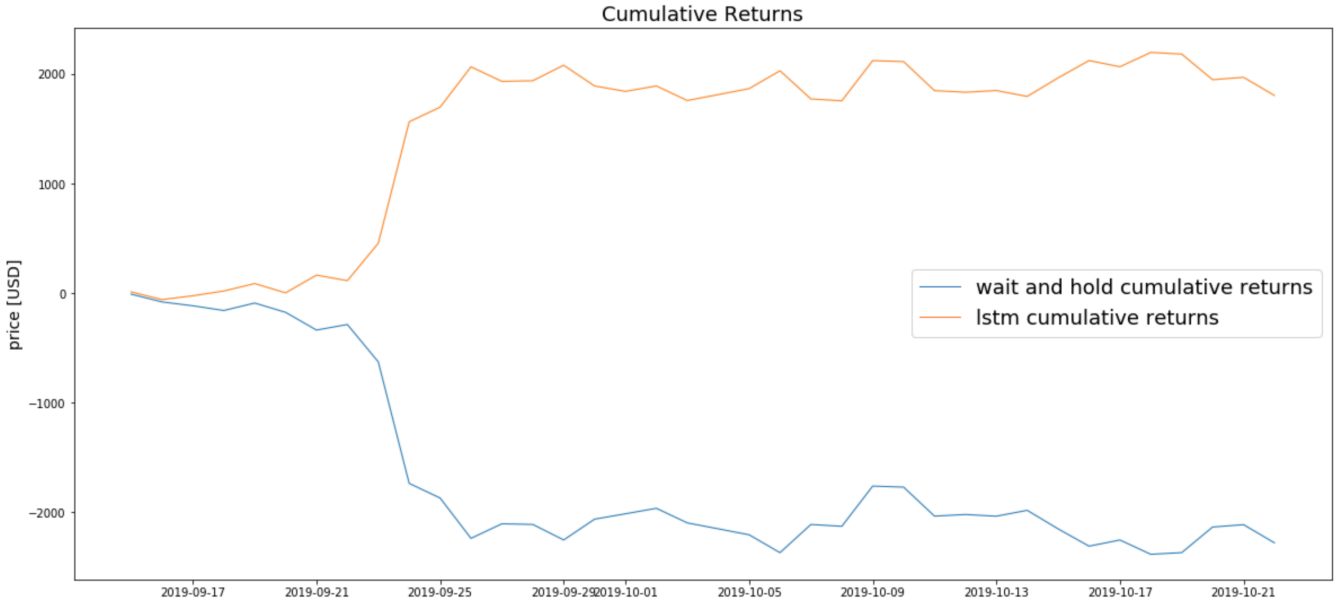
However, we are only interested in the direction of the prediction and not the value itself. Let's look on how the LSTM trading strategy compares to the Wait and Hold trading Strategy.

Justification

Wait and Hold: We have purchased one unit bitcoin on 2019-09-14 when the price was USD 10,358.05. It has experienced a significant fall starting 2019-09-22 form which it hasn't recovered till date. As a result the east and hold strategy has lost us a USD -2,279.85 over the course of validation period.

LSTM strategy: Even though the model price predictions were not very accurate in absolute terms, the direction of the predictions was correct for the most of the time. Over the course of validation period the model has made us a positive profit of USD 1,802.73. Even though the absolute value of the predictions was not very accurate, it managed to get the direction of the price quire well.

The Daily Cumulative returns are shown in the chart below. The blue line represents the returns from the wait and hold strategy and the yellow line is the returns of the strategy based on LSTM classifier.


Cumulative Returns
price [USD]
2000
1000
0
−1000
−2000
2019-09-17
2019-09-21
2019-09-25
2019-09-29
2019-10-01
2019-10-05
2019-10-09
2019-10-13
2019-10-17
2019-10-21
wait and hold cumulative returns
lstm cumulative returns