Bitcoin Price Forecasting using Machine Learning

Submitted by:-

Gnana Teja Peddi
Chirag Khandelwal
Aditi Dhawan
Siddharth Khosla

Bitcoin: A Peer-to-Peer Electronic Cash System

Bitcoin is a cryptocurrency. It is a decentralized digital currency without a central bank or single administrator that can be sent from user to user on the peer-to-peer bitcoin network without the need for intermediaries.

The cryptocurrency is traded by individuals with cryptographic keys that act as wallets. Transactions are verified by network nodes through cryptography and recorded in a public distributed ledger called a blockchain. Bitcoin was invented by an unknown person or group of people using the name Satoshi Nakamoto and was released as open-source software in 2009. Bitcoins are created as a reward for a process known as mining. They can be exchanged for other currencies, products, and services.

Problem

The volatile nature of bitcoin prices often results in bad investments at bad times. Our project aims at developing various models to predict bitcoin prices in order to reduce the risk of loss to investors.

Analysis

1) Our project focuses on predicting the bitcoin prices in the future by training the models on historical bitcoin data.

2) Our models are trained on weekly, monthly, yearly, and daily historical data for a certain period and forecast the bitcoin price for the upcoming period.
3) One can check how much profit or loss one would have borne if one had invested in bitcoin for one of those predicted periods. For example, by training the model on 2015 and 2016 data and predicting the price on a particular day in 2017 one can easily identify how much profit or loss one could have born.
4) For a particular period in June-July, 2018 when there was a steep decline in the bitcoin price, the model could not read the data beyond a point because there were other factors like human emotions, social media activity, etc in play. Our prediction models do not take into account the human emotions that could affect the market, social media tweets or posts, recession, etc. in forecasting bitcoin prices. This is where social media analytics comes into play.

Dataset Exploration and Manipulation

We have exported our data from https://www.coinmarketcap.com where it's freely available to download.
Although our training dataset and test dataset varies based on different models as we've tried to predict the BTC price for various intervals like weekly, montly and yearly. But in most of our models we've used the training data set ranging from 2013 to 2017 and a test data set of November 2017 for a month. The forecasting models are created using the training data set which are used to predict the testing data set.
Source - www.coinmarketcap.com

Dataset

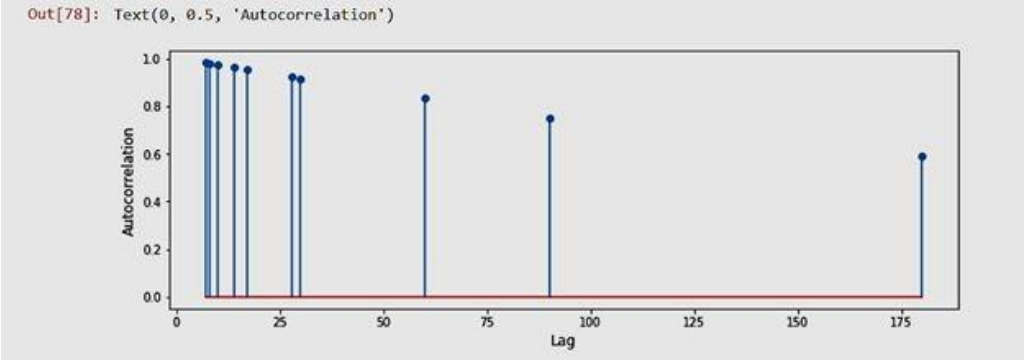
The volume refers to the volume of bitcoin transferred in the market on a particular day and Market Cap refers to the total amount of bitcoins in circulation which is to be capped at 21 million.
Open - Signifies the Opening price of Bitcoin against the USD on that particular day.
High - Signifies the highest price of Bitcoin against the USD on that particular day.
Close -Signifies the closing price of Bitcoin against the USD on that particular day.
Low - Signifies the lowest price of Bitcoin against the USD on that particular day.
Volume - The volume of bitcoin transferred in the market on a particular day
Market Cap - The total amount of bitcoins in circulation which is to be capped at 21 million

Making the data stationary and autocorrelation

Before we move on to developings models, we first need to make the series stationary as it's a prerequisite for any time series based model. Making the data stationary means removing the trend out of it in order to have its own statistical properties (i.e. mean, variance) constant over

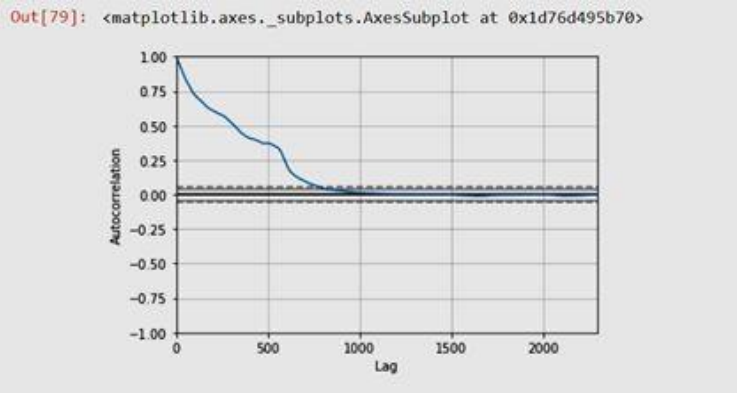
time. This will help the sample to be more predictable by the model since it can be assumed that the statistical properties of the data will be the same in the future as they were in the past. One way of stationarising a time series is through the mehtod of lag differencing, that is difference of two data points within a specified period also known as ‘Lag’.
To find the optimal lag, we use **autocorerlation method.**

Autocorrelation measures the correlation (similarity) between the time series and a lagged version of itself. Below we define varies lag values to see which one causes the highest correlation.



Based on the above chart, where we checked the autocorrelation for different lag intervals like 7, 8, 14, 17, 28, 30, 60, 90, 180, we can conclude that highest correlation occurs at **lag = 7**. It indirectly also means that the data repeats itself every 7 days i.e. a week which is normally observed as BTC price/market corrects itself back on monday after a whole week of up’s & down’s as observed in bitcoin history.

Also, another way of performing autocorrelation is through Pandas tools library using which we can see positive correlation for the first 100 lags.



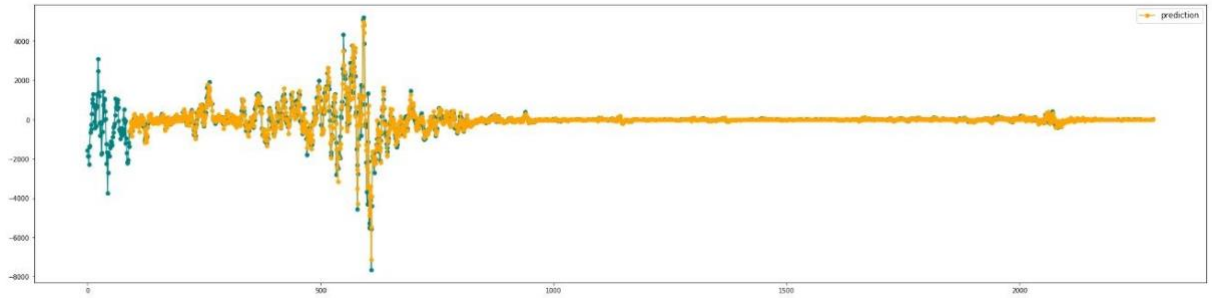
From the above plot, we can conclude that lag of 7 is a good candidate for optimal lag.

Model -1: Linear Regression

We organised the data into a matrix to perform linear Regression on a time series data.

```
print('MAE_day = {0:.3f}'.format(mean_absolute_error(time_series[lag:], lr_prediction)))
print('MAE_week = {0:.3f}'.format(mean_absolute_error(time_series[-7:], lr_prediction[-7:]))) #for last 1 week
print('MAE_month = {0:.3f}'.format(mean_absolute_error(time_series[-30:], lr_prediction[-30:]))) #for last 1 month
print('MAE_year = {0:.3f}'.format(mean_absolute_error(time_series[-365:], lr_prediction[-365:]))) #for last 1 year
```

```
MAE_day = 88.761
MAE_week = 10.437
MAE_month = 5.517
MAE_year = 26.501
```



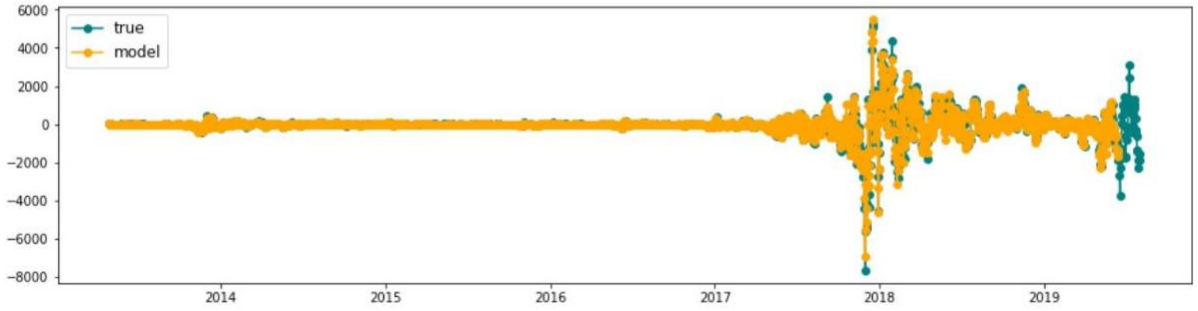
Prediction for previous days	**Mean Absolute Error (MAE)**
Week	10.437
Month	5.517
Year	26.501

As we can see the mean average error is lowest is case of 30 days price prediction

Model -2: Auto Regressive

AR stands for Auto-Regressive time series model where values are modeled as a linear combination of the p past values. Here, we have taken p = 53 using select_order method which finds the MAXLAG for each value of p from 1 to given input (in our case = 100)

```
MAE = 94.702
MAE_week = 10.584
MAE_month = 5.718
MAE_year = 23.783
```



Prediction for previous days	**Mean Absolute Error (MAE)**
Week	10.584
Month	5.718
Year	23.783

Model -3(a): Moving Average

A simple moving average is computed by taking the average price over a certain number of periods. Simple Moving averages are usually constructed using the closing price, while it is also possible to calculate it from the open, the high and the low data points.
Hence, we use moving average (also called rolling mean) to calculate the average of a subset with certain window size and shift forward. Moving average is used to smooth out short-term fluctuations and highlight longer-term trends or cycles.

In this graph, essentially we have plotted price data for 2 year period where the blue line is the original observation and the red, green and yellow lines are forecasted using MA with different windows (3, 7, 14) and we can see that we can still make out general long term but not the accurate fluctuations.

Model - 3(b) : Dual Moving Average Crossover

Essentially we calculate 2 different moving average for the price of bitcoin -

1. Short Term
2. Long Term - will have less variance

The rate of the moving average for both are different but the direction is the same and the different rates of direction induce points where the values of the two moving averages may equal and or cross one another. These points are called the crossover points. In the dual moving average crossover trading strategy, these crossovers are points of decision to buy or sell currency.

Initial investment amount: 1048.89 USD
Maximum invested amount: 1734.64 USD
Current asset value: 22322.46 USD

Hence, using this technical approach - our initial investment would be 1048.89USD and would need to keep investing 1734USD and the final forecasted value would be 22322.46 - which is a pretty decent forecast with an accuracy of 63.45% for a currency that fluctuates immensely.

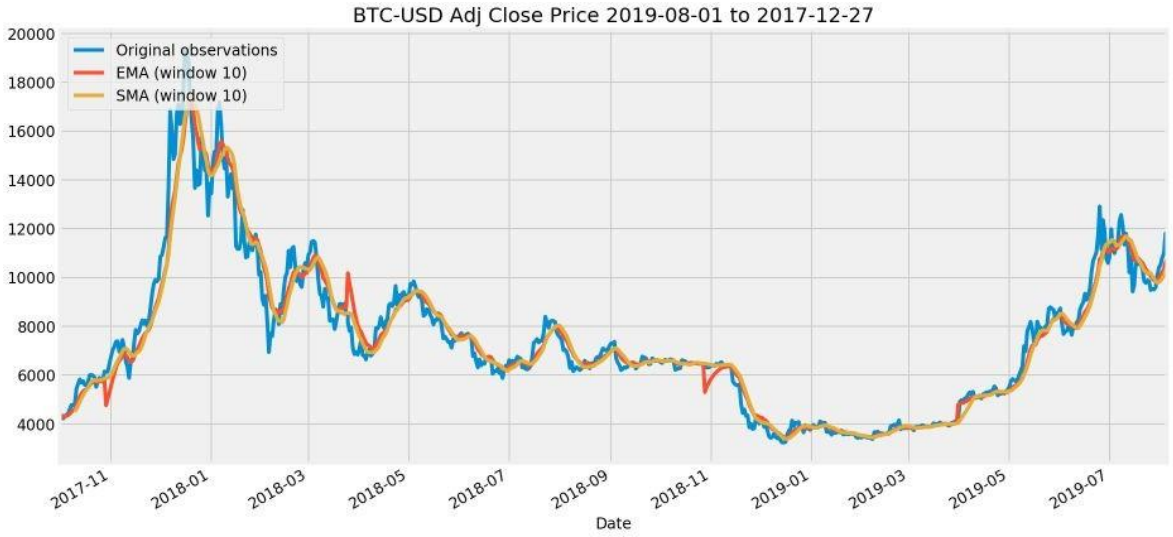
Model - 3(c): Exponential Moving Average

The EMA is a weighted average of the last n (window size) prices, where the weighting decreases exponentially with each previous price/period. I.e the most recent observations are given more weight than the historical observations, unlike Simple Average and that's what makes this model less susceptible to prediction errors since the most recent behavior is more relevant and the historical observations get exponentially less weight.

New forecast = Last period's forecast
+ α (Last period's actual demand
– Last period's forecast)

$$F_t = F_{t-1} + \alpha(A_{t-1} - F_{t-1})$$

F_t	: new forecast
F_{t-1}	: previous period's forecast
α	: smoothing (or weighting) constant ($0 \leq \alpha \leq 1$)
A_{t-1}	: previous period's actual demand



From the graph above - we have drawn a parallel between Exponential moving Avg vs Simple Moving Avg (our 1st model). From the above chart, we can see EMA in red line catches the downwards movement in mid-December faster than SMA in yellow line, and also at the very end, EMA started to catch upwards movement, while SMA still showing downwards trend.

That essentially means that EMA is closer to the real data than SMA which makes sense because - forecasting something that volatile - trend is the key to keep in mind.

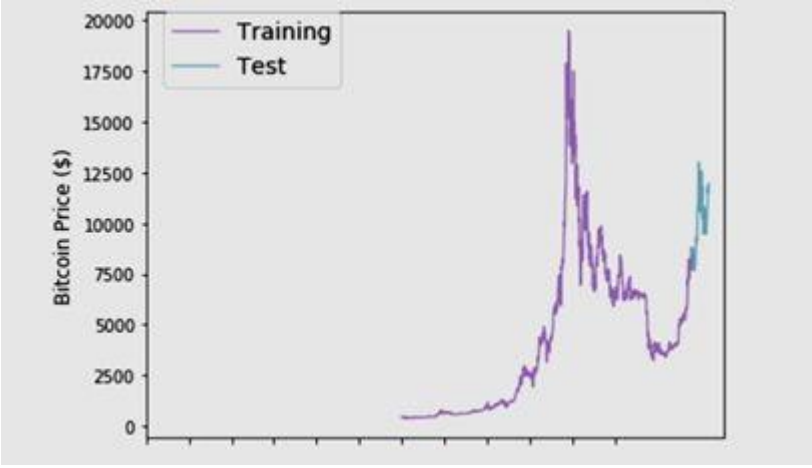
Initial investment amount: 4321.44 USD
Maximum invested amount: 7171.97 USD
Current asset value: 12868.84 USD

With EMA trading strategy, we would have started our investment with 4,321.44 USD, and during the time period of roughly 2 YEARS, we would have needed 7171.97 USD to keep investing, and the final value of our 1 Bitcoin would have been valued at 12868.84 USD.

Model -4 : LSTM - Neural Network

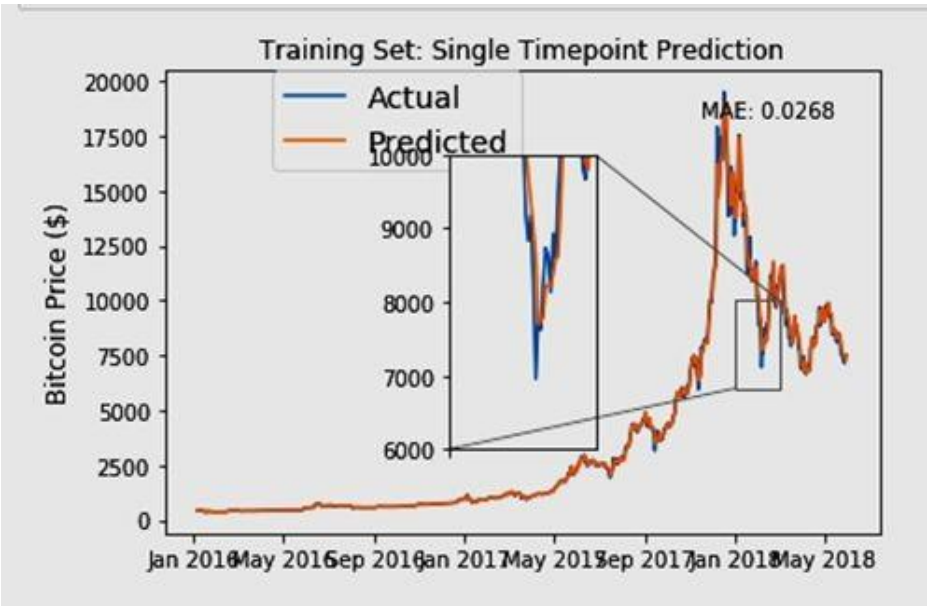
To implement LSTM model, we added two new variables to our dataset -

a) **bt_close_off_high** - gap between the closing price and price high for that day.
b) **bt_volatility** - difference between high and low price divided by the opening price.



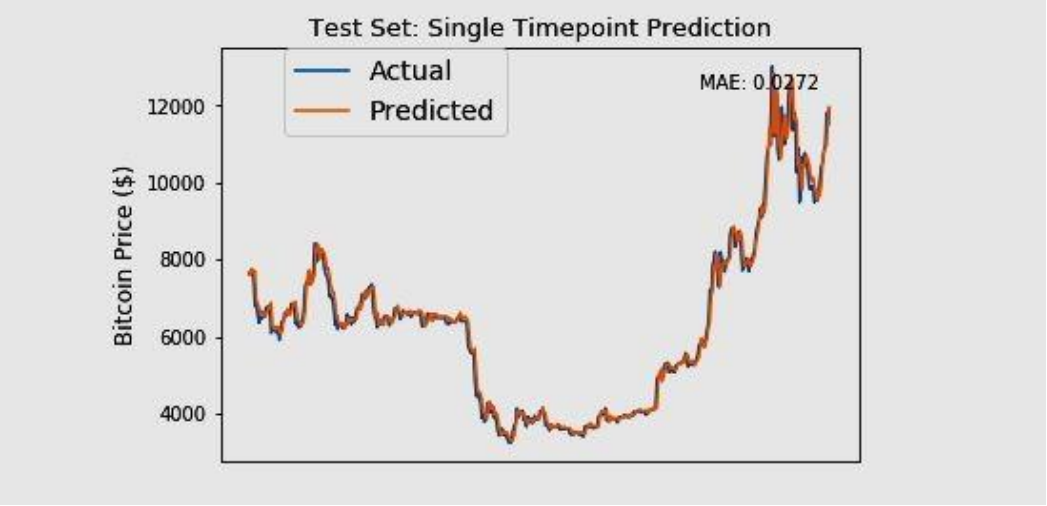
Since we are using a time series dataset, it is not viable to use a feedforward-only neural network as tomorrow's BTC price is most correlated with today's, not a month ago.
Since in this case, we are implementing a recurrent neural network (RNN), we can drop the Date column.

Prediction -
Training Data



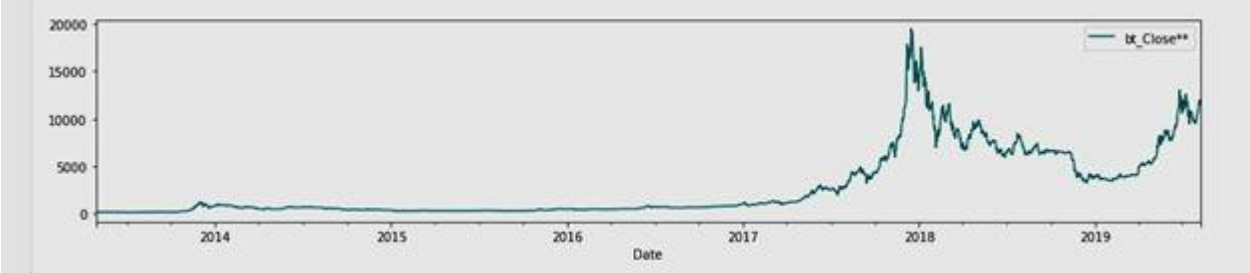
Our LSTM model seems to have performed well on the unseen test set. But it fails to predict a sudden value drop as zoomed in the picture. This can be explained by the fact that our model is based on purely mathematical/logical data (volatility ...), but it doesn't take into account the parameter (this market is highly speculative). In order to improve our model, we should create a model which also includes the sentiment of people (for example: sentiment detection on facebook or twitter)

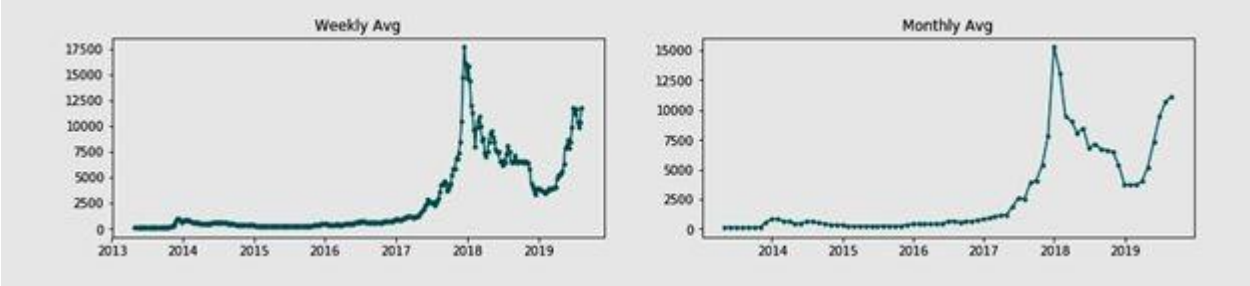
Test Data



Yes, the network is effectively able to learn. But it ends up using a strategy in which predicting a value close to the previous one turns out to be successful in terms of minimizing the mean absolute error.

Model 5: Time Series - ARIMA





After some initial explorations it's now time to create our model. For that we first need to make the series stationary as this is a prerequisite for most of the models. Making the data stationary means taking the trend out of it (de-trending the data) in order to have its statistical properties (i.e. mean, variance) constant over time. This will help the sample to be more predictable by the model since it can be assumed that the statistical properties of the data will be the same in the future as they were in the past. One way of stationarising a time series is through differencing, that is taking the difference of two data points within a specified period, this period is called lag. In order to find the optimal lag we use autocorrelation method.

```
plt.figure(figsize=(10,4))
plt.plot(real_price, "-o", color="teal", label="real price")
plt.plot(bitcoin_forecast, "-o", color="orange", label="predicted")
plt.legend(fontsize=12)
plt.show()
```

Conclusion/Inference

1) Out of all methods LSTM seems like the best fit to the data for a monthly or yearly prediction as gives the closest performance on test and validation data with Mean average error of 0.05 (90 days)

2) With this particular project, some predictions are too good to be true. It means that the network is effectively able to learn. But it ends up using a strategy in which predicting a value close to the previous one turns out to be successful in terms of minimising the mean absolute error.
3) When there was a steep decline in the bitcoin price, the model could not predict beyond a point because there were other factors like human emotions, social media activity, etc. in play affecting the market

References

1) https://towardsdatascience.com/
2) https://en.wikipedia.org/wiki/Bitcoin
3) https://github.com/
4) https://www.investopedia.com/terms/s/stockmarket.asp#ixzz2Ac12urX7
5) https://www.kaggle.com
6) https://www.medium.com
7) https://www.coinmarketcap.com