Bitcoin Price Prediction Using Machine Learning

Department of Computer Science, Drexel University, Philadelphia, Pennsylvania 19104
Mark Klobukov, mk3396@drexel.edu
Greg Matthews, gm453@drexel.edu

***Abstract*— In the past eight years of Bitcoin's history, the economy has seen the price of Bitcoin rapidly grow due to its promising outlook on the future for cryptocurrencies. Investors have taken note of several advantages Bitcoin provides over the traditional banking system. One such trait is that Bitcoin allows for decentralized banking, meaning that Bitcoin cannot be regulated by powerful banks. There is also a market cap of 21 million Bitcoins that can be in circulation, therefore a surplus of Bitcoins cannot be "printed" which would result in inflation. Bitcoin resolves the issue with transaction security by using a blockchain, or a ledger, which records the history of every transaction ever made into one long hexadecimal "chain" of anonymous transactions, which keeps transaction history transparent, but also confidential. Bitcoin as a result has become a very bullish investing opportunity, and due to the huge volatility of the Bitcoin market price, this paper attempts to aid in investment decision making by providing Bitcoin market price prediction. Our team explored several Machine Learning algorithms by using Supervised Learning to train a prediction model and provide informative analysis of future market prices. We start with Linear Regression models, and train on several important features, then build a more efficient Polynomial Regression model. We proceed with the implementation of Recurrent Neural Networks (RNN) with Long Short Term Memory (LSTM) cells. All code is written in Python using Google's TensorFlow software library. We show that the price of Bitcoin can be predicted with Machine Learning with high degree of accuracy. Our team's written code can be found at Github [8].**

***Index Terms*— Bitcoin, cryptocurrency, Linear Regression, Polynomial Regression, Neural Network**

I. INTRODUCTION

The Bitcoin cryptocurrency experienced tremendous growth over the past year. The price of one Bitcoin went from about $750 at the end of 2016 to over $10,000 at the end of 2017 [1]. Great rates of growth were also observed in the other cryptocurrencies, such as Etherium and Litecoin. There is currently a shortage of quantitative analysis tools and techniques for predicting prices of cryptocurrencies. Mathematical analysis has a well-established place in the financial industry for evaluating expected returns of a stock of a given company or performance of an entire portfolio. However, Machine Learning literature is lacking verification of whether the stock analysis techniques are valid for the cryptocurrencies, and if so, how they can be modified. That is, what features need to be removed or added as a basis for price prediction, whether current Machine Learning algorithms work for cryptocurrencies, and which approach yields the best results.

In this paper, we investigate these questions. Such analysis is relevant given a great amount of attention that cryptocurrencies, in particular Bitcoin, are generating. Both individuals and large financial firms are attracted to cryptocurrencies because of the transparency and anonymity that they provide to their users, as well as their resistance to fraud due to the distributed nature of the ledger records. Moreover, purchasing cryptocurrencies is promising in terms of making profit, and should be of interest to investors. In addition to familiarizing themselves with industry trends and political and economic news, they can utilize Machine Learning models to help them make a decision to buy or sell cryptocurrency.

As described in greater detail in the future sections of this paper, Machine Learning algorithms can in fact be applied to cryptocurrency data to predict future price movements. Different approaches produce results of different accuracies. However, given a strong correlation between actual direction of price change and prediction of the algorithms, we were able to show that mathematical analysis can aid investors' decision-making considerably and allow one to increase chances of making profit by trading cryptocurrencies.

Quantitative prediction of the prices of cryptocurrencies can also be utilized to build an autonomous agent that performs trades on behalf of the investor based on historical price information, current news, sentiment analysis, and real-time data. Our work only evaluates Machine Learning models trained on historical data, but it can be extended to building an entire trading system similar to those that are already being used for trading financial securities.

In section II of this paper, we provide a background overview on current prediction modeling techniques being used by stock exchanges, and follow up with several differences as well as similarities that can be made by treating cryptocurrencies as one would a share of stock. We then proceed by stating how cryptocurrency prediction modeling is a relatively young technique that has not been fully explored in research literature.

Section III contains a description of the approach we followed to ascertain whether the price of Bitcoin can be

predicted with Machine Learning. In particular, we discuss the application of Linear Regression, Polynomial Regression, and Recurrent Neural Networks (RNNs) with Long Short Term Memory (LSTM) cells to the task of predicting the market price of Bitcoin. We identify relevant features for price prediction and determine accuracy of single- and multi-feature Machine Learning models.

In Section IV, we show and evaluate results of the experiments performed on the historical price of Bitcoin by comparing the actual cryptocurrency price in the past with the predicted values. Section V provides an overview of relevant topics in research literature with regard to using Machine Learning in the financial industry. Finally, Section VI presents concluding remarks and suggests direction for future work on the subject of applying Machine Learning to cryptocurrency price prediction.

II. Background

Due to the high volatility associated with the market price of Bitcoin, prediction models have become more and more challenging for investors to accurately forecast investment decisions. Thinking in regards to a stock market investment, a typical approach many investors would use as a prediction metric, is called Fibonacci retracement. Fibonacci retracement helps identify the opportune moments for traders to buy or sell stock at the optimal price. Fibonacci retracement utilizes the volatility of the stock, meaning that the two extreme trends in a graph are found, and the calculated vertical distance is then divided by key Fibonacci ratios, most commonly the golden ratio of 0.618. These values are then used to help identify the most prominent support and resistance levels [10].

The problem with such prediction models however, is that the Fibonacci retracement levels are static, and are constant calculated values that are only good metrics for simple identification. In regards to Bitcoin, time series predictions is not a novel idea for brokerage companies, and many brokers have invested heavily to improve performance of forecasting stock prices and foreign exchange rates by way of Machine Learning. Bitcoin, a cryptocurrency first launched in 2009 by Satoshi Nakamoto, has parallels with many of the stock markets such as the New York Stock Exchange (NYSE) and can be modeled in similar ways.

However, there are key differences between Bitcoin and the stock market which makes our prediction model much different in nature. For one, Bitcoin commands a much smaller transaction volume as compared to many of the major stock exchanges. The average daily turnover for Bitcoin at the present day is $21 million, whereas trillions are seen for the latter. As a result of the small transaction volume, lower liquidity, and huge peak in investment turnover of Bitcoin as the future of decentralized banking increases, the market price of Bitcoin has become highly volatile and as a result, challenging to make successful predictions [1].

While there is a great amount of scholarly literature and practical guides on Machine Learning applied to stock market, there is a lack of information about predicting movement of the value of cryptocurrencies. Namely, there is no consensus on the features that are to be used for training a prediction model. Our research primarily focuses on blockchain size as an important feature of the market price trend, using various Machine Learning models, as well as expanding our input variables into multi featured data. Previous attempts at time-series forecasting have been well-documented and there are a lot of resources and tools available to use for free which our team plans to utilize in order to solve the problem at hand efficiently rather than reinventing the wheel at each stage. Specifically, Quandl API is used as the primary distributor of large feature data, and Google's TensorFlow library for Python allows for function calls to complex Machine Learning algorithms such as Gradient Descent. Other libraries such as Keras will also be implemented for more complex modeling.

III. Approach

To begin exploration into Bitcoin market price prediction, our team implemented Linear Regression, a Supervised Machine Learning algorithm that outputs a linear prediction model. Linear Regression works by reducing the squared mean error of a proposed hypothesis $h_\theta(x)$ of what the actual output $y$is, which for our project is the market price of Bitcoin. The hypothesis function $h_\theta(x)$ is computed by multiplying the calculated matrix of "weights" called Θ, taking its transpose and performing matrix multiplication on x, which is a matrix of features such as the Blockchain size, Number of Bitcoin wallets, or Bitcoin Hash Rate. These features are vectors of known historic data that will be used to train our model to predict future prices given specific feature values [2].

$$h_\theta(x) = \Theta_0 + \Theta_1 x_1 \ldots + \Theta_n x_n = \Theta^T x \qquad (1)$$

To calculate the mean squared error of our hypothesis, or more formally the cost, we take the difference of our hypothesis with the actual output y , and square the value. This result is then summed over all the training examples m of every feature n in x.We then normalize by dividing by two times the number of training examples m. In the case of one feature, the following cost function is shown [2].

$$J(\Theta) = 1/2m * \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 \qquad (2)$$

Now that the cost function has successfully been computed, we have to iteratively reduce the mean squared error of our hypothesis by continually updating the matrix of weights Θ until they converge to a global minimum value. This is performed by either using the Normal Equation, but for our purposes we used Gradient Descent which is more efficient for larger data. This is achieved by taking the partial derivative with respect to Θ of our cost function, multiplying it by a learning rate α,whereby Θ will converge faster or slower depending on the chosen value of α,, and subtracting this value from the current weight Θ. One can think of taking the

partial derivative as finding what direction to move the current $\Theta_j$value by, and increasing or decreasing the speed at which the process is performed by the learning rate α.The process is shown mathematically below [2].

$$\Theta_j := \Theta_j - \alpha \frac{\delta}{\delta \Theta_j} \quad (3)$$

$$\Theta_j := \Theta_j - \alpha/m * \sum_{i=1}^{m} (h_\Theta(x_i) - y_i)x_i \quad (4)$$

To solve the underfitting problem of Linear Regression, Polynomial Regression was introduced. Working in much the same way as Linear Regression, Polynomial Regression extends our prediction model to take on higher powers, and thus improve prediction accuracy and reduce mean squared error. Polynomial regression allows the prediction model to fluctuate and "curve" to the data more fluently. This is done by adding more features of the same input x, but with increasing power. Mathematically, the hypothesis function now becomes [3]:

$$h_\Theta(x) = \Theta_0 + \Theta_1 x_1 + \Theta_2 {x_1}^2 + \Theta_3 {x_1}^3 \ldots \quad (5)$$

The following mathematical equation is specifically for one feature x_1 but can also be expanded to n features. The expansion to the degree d must be tested as to not cause overfitting of the data, which can result from our model being overly sensitive to our training data, and perform suboptimally for newer unseen data. Another important factor to take into account, is scaling our features for them to have normalized means. If we have multiple features and one of them ranges in value from [0, 1000], while another is from [0, 0.1], then there will be higher weight associated to the higher scaled feature. Convergence of the cost function using Gradient Descent will take much longer as well due to the variance in feature values. To resolve this issue, we perform mean normalization and feature scaling on all our input data, as shown [3]:

$$x_i := \frac{x_i - \mu_i}{s_i} = \frac{x_i - mean(x_i)}{std(s_i)} \quad (6)$$

By subtracting the mean of x for all data points in $x$and dividing by the standard deviation, we allow all features to vary by the same value of roughly -1.5 to 1.5.

The third implemented strategy for Bitcoin price prediction is using Recurrent Neural Networks (RNNs) with LSTM cells. Multi-feature RNNs with Long Short Term Memory, or LSTM, were developed to further improve prediction accuracy. Due to the RNN's backpropagation feature, they are one of the best models which are often used for time-series prediction problems as they store the state of features from previous time-steps in memory and exploit them to predict the future [9]. However, it has been proven time and again that RNN's performance drops when RNN have to remember information for long periods of time. LSTMs are special types of networks which solve the long-term dependency problem by means of a 'cell state'. The cell state allows for easy propagation of information across the entire chain.

IV. EVALUATION

Finalizing the results of Linear Regression, Fig 1. shows the result of performing Linear Regression using the day as a feature to predict the market price of Bitcoin over the last 365 days. Learning rate $\alpha$was set to 0.03, and Gradient Descent was performed over 10,000 iterations. The cost function converged to a mean squared error of $1039.58.

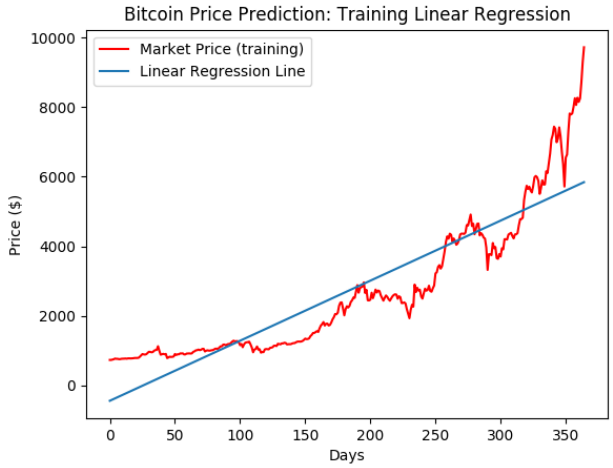

Fig 1. Linear Regression model given last 365 days, using training rate of 0.3 and 10,000 training iterations

Linear Regression has showed us that a simplified model of Bitcoin market price can be produced with minor complexity details, albeit larger error due to the constraint of the regression line being linear. As the price of Bitcoin exponentially begins to increase, the error of Linear Regression will continually get worse as a result of underfitting. Moving on to the second Linear Regression model, where Blockchain size is used as our main feature, we begin to find some striking similarities.

Blockchain size was used as a good representative feature for our model because increasing Blockchain size correlates to more investors making Bitcoin transactions, and Bitcoin market price is heavily weighed by consumer demand and interest. Shown in Fig 2 is the Linear Regression training output using Blockchain size as our feature with learning rate α at 0.3, and 10,000 iterations of Gradient Descent. After training our model, the cost function converged to a mean squared error of $1053.29, almost the same as the previous date feature. This shows that date and Blockchain size have a dependency and strong correlation, therefore both features should not be used simultaneously when performing multi-feature modeling due to increased bias of theta weights when using features that vary in similar ways.

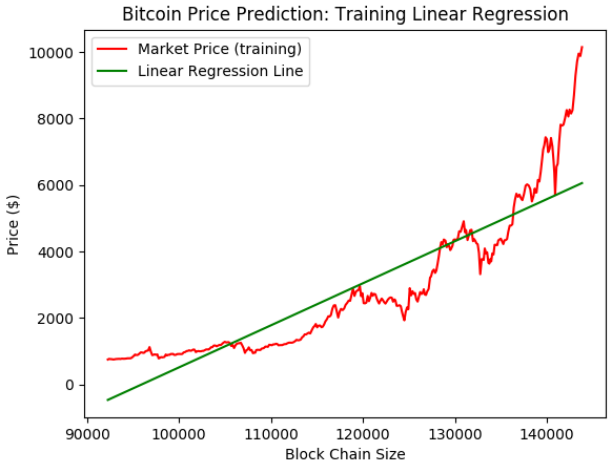

Fig 2. Linear Regression model given Blockchain size, using training rate of 0.3 and 10,000 training iterations

Moving into Polynomial Regression, our results show a dramatic decrease in mean squared error as compared to Linear Regression. Fig 3 shows Polynomial Regression with a degree d of 4, transforming our hypothesis function into a quartic model. The learning rate α is kept unchanged at 0.3, and 20,000 iterations of Gradient Descent are performed. The cost function converged to a mean squared error of $574.80, a factor of 1.84 less than Linear Regression.

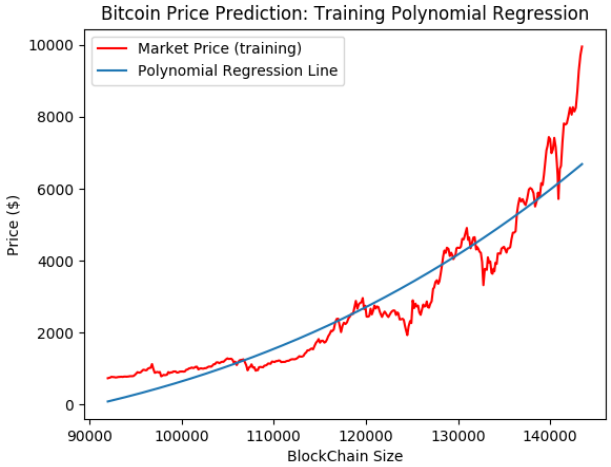

Fig 3. Polynomial Regression model given Blockchain size, using training rate of 0.3 and 10,000 training iterations with degree of 4

Improving our Polynomial Regression model further, we increase the degree d from a quartic model, to a power of 50. The outcome of our Polynomial Regression model improves even further, with our cost function converging to a mean squared error of $189.95, a factor of 5.6 improvement over the standard Linear Regression model, and a factor of 3.04 improvement over the quartic Polynomial Regression model.

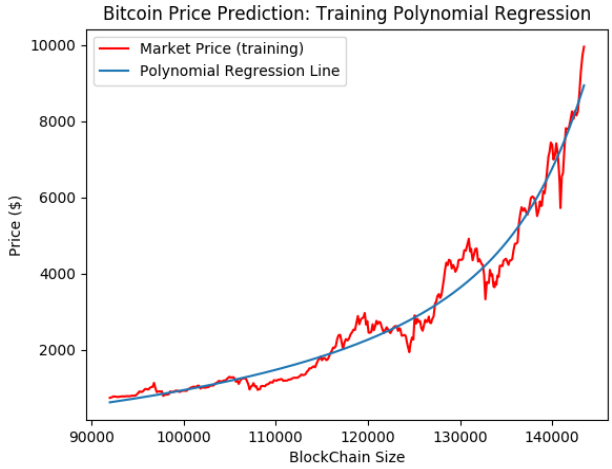

Fig 4. Polynomial Regression model given Blockchain size, using training rate of 0.3 and 10,000 training iterations with degree of 50

Evaluating all Linear and Polynomial Regression models explored, Polynomial Regression with degree d of 50 gave us the best results in terms of reducing the mean squared error of our training set, while also not overfitting to the data. The model was not affected by minute noise in the market price training set, which means it is generalizing to the data well, resulting in an improvement in accuracy given new unforeseen future feature data.

The third and final model that we trained is the RNN with LSTM cells. The model was trained on the number of transactions per day, hashing rate, number of Bitcoin users per day, and transaction volume for every day over the past 3 years (except the last month, which was set aside for testing). Before the input features were supplied to the model, an additional feature selection optimization was applied. This was done to further reduce the feature set by removing input features from the model which had little correlation with the output. The feature selection method used was Reduced Feature Elimination (RFE). The idea behind RFE is straightforward. A simple model (the Linear Regression model in this case) is used as training set with output values already known. Features are recursively removed and a model is built with the attributes remaining. Once the process is complete, features which do not significantly contribute to predicting the price of Bitcoin for the future, can be removed.

After RFE was implemented (using a built-in function in the Scikit-learn library), the number of features was reduced from six to four. After feature selection was done, the LSTM model was tuned. To prevent overfitting, the number of neurons in the hidden layer had to be specified. Since there is no determined rule for picking the perfect number of neurons, the following popularly accepted rule of thumb was used:

$$\frac{N_{sample}}{\alpha(N_{input}+N_{output})} = \frac{1065}{3*(4+1)} = 71 \qquad (7)$$

where α is a range of values from 2 to 10. For this case, α value of 3 was taken (the lower the value of α, the less the network overfits). Applying the above formula yielded a value close to 71.

The number of epochs was chosen empirically to be 100. It can be seen in Fig. 4 that by the time the epoch count becomes 100, both the training and test sets' loss values have plateaued and almost converged. This implies that the model was fitted.

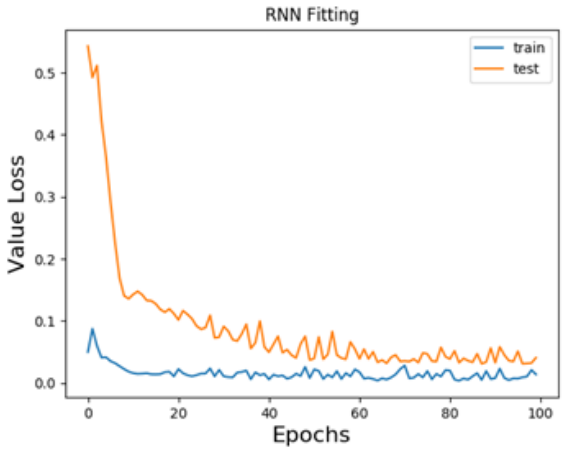

Fig 4. Value of the loss function for testing and training data sets vs the number of epochs

The model was trained on the data prior to November 2017 and tested against the actual prices during November 2017. Fig 5 shows a plot of actual and predicted Bitcoin prices.

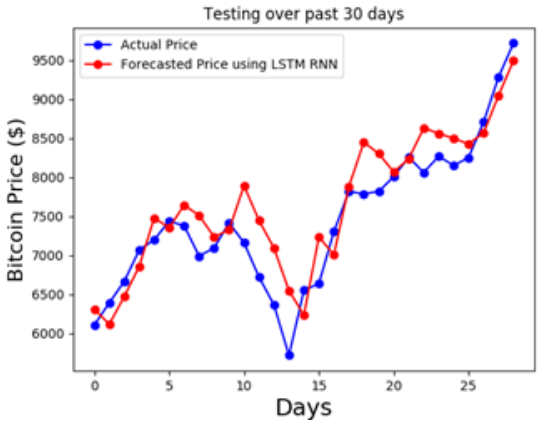

Fig 5. Actual vs predicted price of Bitcoin during November 2017

The following table shows comparison of average prediction accuracies of the algorithms implemented in this project.

Table 1: Bitcoin Price Prediction Accuracy over 15 Days

Machine Learning Model	**Accuracy (%)**
Linear Regression	69.9
Polynomial Regression	96.7
Recurrent Neural Network With LSTM	96.2

Given the percent accuracy results shown in Table 1, we see a percent accuracy of 69.9% when using the Linear Regression Model. This can be attributed to the growing error when attempting to fit a linear model onto a very noisy, nonlinear data set. Moving to the Polynomial Regression model, we saw the best performance with a percent accuracy of 96.7%, which shows that increasing the polynomial degree of the hypothesis function greatly reduces the cost function. The Recurrent Neural Network model incorporating LSTM cells performed slightly less accurate than Polynomial Regression, having an accuracy of 96.2%. In regards to long term prediction making, we see Polynomial Regression begin to outperform the RNN model, which is more optimal for short term price predictions. Referring to Fig. 5, we see how the RNN model is sensitive to fluctuations of the bitcoin price and performs well in the short time frame.

V. Related Work

Several attempts at predicting price movements of Bitcoin are found in research literature. Devavrat Shah and Kang Zhang of MIT developed a trading technique in 2014 based on predicting Bitcoin price with Bayesian regression on a latent source model [4]. The latent source model provides a way to make a binary classification. In the case of this research, the binary decision was that either the Bitcoin price will go up or down based on time-series data. The researchers combined this technique and Bayesian regression, and predicted Δp, the price change in the next 10 seconds. For Δp higher than a threshold value *t* they bought 1 Bitcoin; for Δp lower than -*t*, they sold 1 Bitcoin. This approach allowed Shah and Zhang to achieve a 89% return in 50 days [4]. These results demonstrate that even a simpler approach such as Bayesian regression with no data other than historical price can be useful as an investment aid or even as a basis for an autonomous trading agent.

In another study by the researchers at the Athens University of Economics & Business, Bitcoin prices were predicted based on sentiment analysis and econometric data [5]. Since the price of Bitcoin does not necessarily conform to classical economic influences of supply and demand but is significantly influenced by the public opinion, the research team obtained data from Twitter posts, number of Google and Wikipedia queries, and the hash rate on a daily basis over the course of 78 days. The data was used as an input to a Support Vector Machine (SVM) algorithm for classification of sentiments. In addition, various economical variables were explored and correlated with Bitcoin price data. They include USD/EUR exchange rate, value of the S&P 500 index, and number of Bitcoins in circulation.

The researchers concluded that the price movements of Bitcoin are positively correlated with the number of Wikipedia views, hashing rate, and sentiment ratio from Twitter. They also showed negative correlation between price of Bitcoin and USD/EUR exchange rate as well as S&P 500 index [5]. No autonomous agent was built, and no specific future Bitcoin

price prediction was made. However, the study successfully demonstrated correlations between the price of Bitcoin and other easily obtainable data, and these conclusions can be used to build an actual trading system.

The literature is lacking Bitcoin-specific publications on using LSTM networks to make a prediction. However, there are some recent publications on application of LSTMs to stock market predictions. While these approaches use features that are different from the ones that are directly related to Bitcoin, they are still worth examining.

A team from the Federal University of Minas Gerais published a paper in which an algorithm for stock price prediction with LSTM networks is suggested [6]. The researchers collected price data from 2008 to 2015 for several Brazilian stocks and trained a Recurrent Neural Network with LSTM cells to do binary classification on each stock. The classification tells whether the stock price will go up or down in the next 15 minutes. This project did not yield promising results: the team concluded that predictions achieved a 55.9% accuracy, which is slightly better than flipping a coin. This shows that working with LSTMs is not a trivial task, and addition of multiple features to the training dataset is likely necessary to achieve sufficient accuracy.

Research is also being carried out in the area of applying deep neural networks (DNNs) to predicting stock prices. Matthew Dixon *et al.* from the Illinois Institute of Technology created an automated stock trading system based on 5-minute interval prices from 1989 to 2013. To accommodate a large amount of data, feed-forward topology was used instead of the more popular back propagation [7]. The implemented DNN consisted of four layers. The model made positive, negative, or neutral prediction on a stock and make a trade if the magnitude of expected change was higher than a threshold value. The system achieved a prediction accuracy of 42% with a standard deviation of 11%. In the best case, the model was able to make a classification with a 68% accuracy for one of the stocks. The results in the best case are better than in the research in [6] but worse on average.

Based on the aforementioned research projects, it can be concluded that the future direction of work on predicting prices of stocks and cryptocurrencies has to take multiple metrics and features into account. Econometric data and sentiment analysis were only incorporated into projects that used simpler algorithms, such as Bayesian Regression. However, the projects involving DNNs or RNNs with LSTMs were trained on the datasets containing only the historical prices of stocks. This attests to the novelty of our work: we used multiple features in training the Neural Network, which is something not found in the research literature. Incorporating multiple features into prediction methods with these types of Neural Networks has a potential to produce more accurate results.

VI. Conclusion

In this paper, several approaches for Bitcoin price prediction were investigated. We compared the results of prediction with Linear Regression, Polynomial Regression, and Recurrent Neural Networks with LSTM cells. While the applications of the first two methods have been documented in the research literature and in practice, our LSTM method is novel. The research contribution of this technique is that we predicted a numerical value of price instead of performing a binary classification, as well as used multiple features to train the model. The LSTM method performed notably better than the other two approaches, and we believe that further research on using Neural Networks for time-series prediction is very promising to financial data analytics and other fields.

Our work can be extended further using some of the approaches described in the Related Work section. Namely, the LSTM-based model can be used as a part of the autonomous trading agent like the one in [4]. It is worth investigating the scalability of our proposed approach. In particular, the important questions for further research are how far into the future should the price be predicted, and how many Bitcoins should be traded at a time by the autonomous agent.

The LSTM model, as well as the autonomous agent based on it, can be further enhanced with sentiment analysis as in the project by Georgoula *et al.* [5]. Historical sentiments from Twitter, number of search queries from Wikipedia and Google, and other metrics reflecting public interest in Bitcoin can be used to influence the weights during model training. Moreover, the current sentiments can be combined with the prediction of the LSTM model to influence an autonomous trading agent's decision whether to buy or sell Bitcoins at a given moment of time.

VII. References

[1] B. Reutzel, D. Palmer, M. Hochstein, O. Godbole, D. Palmer and , "Bitcoin Price Index - Real-time Bitcoin Price Charts", *CoinDesk*, 2017. [Online]. Available: https://www.coindesk.com/price/. [Accessed: 02- Dec- 2017].

[2] A. Ng, "Linear Regression With One Variable", Stanford, CA, 2017.

[3] A. Ng, "Linear Regression With Multiple Variables/Polynomial Regression", Stanford, CA, 2017.

[4] D. Shah and K. Zhang, "Bayesian regression and Bitcoin", *Massachusetts Institute of Technology*, 2014.

[5] I. Georgoula, D. Pournarakis, C. Bilanakos, D. Sotiropoulos and G. Giaglis, "Using Time-Series and Sentiment Analysis to Detect the Determinants of

Bitcoin Prices", 2015.

[6] D. Nelson, A. Pereira and R. de Oliveira, "Stock Market's Price Movement Prediction With LSTM Neural Networks", in *Neural Networks (IJCNN), 2017 International Joint Conference*, 2017.

[7] M. Dixon, D. Klabjan and J. Bang, "Classification-based Financial Markets Prediction using Deep Neural Networks", *Illinois Institute of Technology*, 2017.

[8] G. Matthews and M. Klobukov, "Gregory-Matthews/Bitcoin-Price-Predictor", *GitHub*, 2017. [Online]. Available: https://github.com/Gregory-Matthews/Bitcoin-Price-Predictor/. [Accessed: 02- Dec- 2017].

[9] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] C. Murphy, "What is Fibonacci retracement, and where do the ratios that are used come from?", *Investopedia*, 2017. [Online]. Available: https://www.investopedia.com/ask/answers/05/fibonacciretracement.asp. [Accessed: 03- Dec- 2017].

Appendix A:

Test Dataset: Predicted vs. Actual Bitcoin Price

Date	Linear Reg. ($)	Polynomial Reg. ($)	RNN (LSTM) ($)	Market Price ($)
11/30/17	6036	9174	9510	9879
11/29/17	6013	9027	9581	9953
11/28/17	5992	8897	9792	9718
11/27/17	5970	8761	9168	9284
11/26/17	5949	8641	8686	8707
11/25/17	5929	8521	8637	8251
11/24/17	5909	8409	8647	8149
11/23/17	5888	8292	8591	8268
11/22/17	5871	8201	8805	8060
11/21/17	5848	8077	8373	8256
11/20/17	5826	7959	8208	8008
11/19/17	5805	7850	8315	7817
11/18/17	5787	7759	8430	7787
11/17/17	5765	7651	7947	7815
11/16/17	5745	7555	7147	7301