Short Term Crypto Currency Price Predictor using ML

Submitted by –

Aniket Mittal 20BCS6819

Bachelor of Engineering
In
Artificial Intelligence and Machine Learning

Under the Supervision of -
Kushagra Agrawal (E13465)

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Chandigarh University, Gharuan, Mohali – 140413,

Punjab

Short Term Crypto Currncy Price Predictor using Machine Learning Models

Aniket Mittal
Apex Institute of Technology
Chandigarh University
Mohali, India
20BCS6819@cumail.in

Kushagra Agrawal
Professor, AIT-CSE
Chandigarh University
Mohali, India
Kushagra.e13465@cumail.in

***Abstract*—Cryptocurrency has grown significantly in recent years. Additional progress in the planetarium has recognized the importance of embracing quantitative benefits and rapid progress in this field. In today's financial markets, the decision to buy or sell cryptocurrency is an interesting challenge that traders face every day. During the year, it reached unprecedented highs, leading to the idea that explains the growth trend. The question of whether the movement of financial assets can be predicted has been of great interest to investors, economists and researchers in recent years. Therefore, the paper uses machine learning to build a model to predict the price of Cryptocurrency using technical indicators, which is the most important to learn market trends. This study explores how to adapt Long Short-Term Memory (LSTM) to build cryptocurrency price prediction models. The main factors used are the available price, closing price, high price, low price, volume and market capitalization among several cryptocurrencies, based on the size of important trading characteristics that affect the unpredictability of using the model to improve the efficiency of the process. However, the cryptocurrency market lacks a strong and unpredictable regulatory structure, making price prediction more difficult and complex. From the analysis, it was found that the machine learning model provides better performance in cryptocurrency price prediction.**

In this project, we use LSTM version of recurrent neural network, price for Bitcoin. To better understand the price impact and make an overview of this great invention, we first take a brief look at the Bitcoin economy. Next, we define a database that includes data from the market index, sentiment, blockchain. In this analysis, we show the use of LSTM structure and the aforementioned time. The purpose of this research paper is to derive an algorithmic model with high prediction accuracy for the price of Bitcoin the next day through random forest regression and LSTM and explain the variables that affect the price of Bitcoin. There is more literature on Bitcoin price forecast research and research methods mainly based on time series ARMA model and deep learning LSTM algorithm.

Keywords— Bitcoin, Crypto Currency, Machine Learning, Blockchain, Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), Prediction

INTRODUCTION

Bitcoin and other crypto currencies are a decentralized digital currency that uses cryptography for security and is not controlled by governments or financial institutions. It was created in 2008 with a paper entitled "Bitcoin: A Peer-to-Peer (P2P) Electronic Cash System" by an individual or a group of individuals using the pseudonym Satoshi Nakamoto (2008). Bitcoin transactions are recorded on the public blockchain, which allows anyone to see the history of a particular Bitcoin. The decentralized nature of Bitcoin allows it to be used independently of central banks and can be instantly transferred worldwide. It became popular as a medium of exchange and store of value. In the last 10 years, in November 2021, one coin has exceeded USD 68,000, and the total value once exceeded USD 1.2 trillion.

However, Bitcoin as a commodity has high volatility. During the seven-year period from April 2015 to April 2023, Bitcoin's average daily return was 3.85%, which is 2.68 times the return rate of gold in the same period, 3 times the S & P500. This is 36 times higher. Bitcoin's functionality as a commodity, as a store, and as a currency has been called into question due to its large price swings.

Taking advantage of Bitcoin's security and decentralization, it has become a challenge how to understand the trend of Bitcoin in order to reduce the risk of Bitcoin floating. Many researchers try to understand the trend of Bitcoin through the correlation between the price of Bitcoin and the price of other commodities.

In past studies, another form of research to understand the Bitcoin price trend is to predict the price of Bitcoin in the future using AI algorithms and powerful computing power of computers. Machine learning technology has become a hot research area in the 21st century with improvements in hardware performance. Initially, machine learning was used in various fields such as stocks, crude oil market, gold market and futures market.

Predicting AI by Bitcoin mainly falls into two categories. The first category is classification research to predict whether Bitcoin will rise or fall in the future. Standard errors of DA and F1. Another category is the regression test on Bitcoin price prediction, with relative error RMSE and MAPE. Because the price of Bitcoin fluctuates so much, just knowing whether the price of Bitcoin will rise or fall in the future will not allow investors to avoid risk. Instead, it is more useful to take the price of bitcoin as a reference value.

Predicting AI by Bitcoin mainly falls into two categories. The first category is classification research to predict whether Bitcoin will rise or fall in the future. Standard errors of DA and F1. Another category is the regression test on Bitcoin price prediction, with relative error RMSE and MAPE. Because the price of Bitcoin fluctuates so much, just knowing whether the price of Bitcoin will rise or fall in the future will not allow investors to avoid risk. Instead, it is more useful to take the price of bitcoin as a reference value.

Based on the need to avoid price risk as Bitcoin, this study chooses a machine learning random forest regression algorithm and a neural network algorithm LSTM model to predict Bitcoin's price. I mainly focus on the performance of random forest regression in predicting Bitcoin price while using LSTM prediction results as a comparison. Random forest regression is a variant of random forest regression. Unlike black-box neural network technology, random forest regression, like machine learning, can provide the value of each explanatory variable in predicting Bitcoin through the results of weak learners.

LITERATURE SURVEY

We're all predicting where bitcoin spending will be in one year, two years, five years, or 10 years. Waiting is hard, but we all love to do it. Buying and selling bitcoins can be done accurately every time. It has proven to be an asset for many people in the past and is still earning a lot of money. But it won't hurt either. If it is not properly thought and considered correctly, you can lose a lot of money. You must have a great understanding of how and exactly the price of bitcoin changes (organic market, trends, news, etc.), which means you must understand how we make bitcoin predictions. With this in mind (supply and demand, regulations, news, etc.), bitcoin technology and its development should be considered. In addition to this, we have to deal with the technical aspects of using various algorithms and technologies that can accurately predict the price of bitcoin. We have found some models that exist today, such as the biological nervous system. (BNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Automatic Regressive Integral Moving Average (ARIMA) etc. Time series is usually a series of numbers over time. This is because, as a time series data set, the total data set must be divided into two parts: input and output. In addition, LSTM is excellent compared to classical statistical linear models because it can easily solve multi-input prediction problems.

METHODOLOGY

Machine learning is an important branch of artificial intelligence (AI). Depending on whether or not there is a target variable, it can be classified into supervised training, unsupervised training, and strengthening training. The purpose of this study is to predict the future price of Bitcoin, so the regression function and supervised learning are used. The logic of the integration of machine learning is that after the algorithm has been determined, a learner is generated and the learner-high accuracy is obtained by repeatedly training the learner through the training data and validation process. Finally, tests replace students who are trained to evaluate and use data.

The random forest regression and LSTM model training in this paper were performed using the Python open-source machine learning library. sklearn, the library used by Random Forest Regression, uses Keras for LSTM search. Pre-processing and data collection is done by panda.

Random forest is a type of ensemble of multiple regression trees. The advantages are clear, but the predicted results are limited to study samples. The principle of the regression tree is to divide the main group into subgroups using a certain variable index, and the classification is based on minimizing the average squared residuals of each group shown in the equation below:

$$\frac{1}{n_1}\sum_{i=1}^{n_1}\left(y_i - \overline{y_{(1:n_1)}}\right) + \frac{1}{n_2 - n_1}\sum_{j=n_1+1}^{n_2}\left(y_j - \overline{y_{(n_1+1:n_2)}}\right) \rightarrow min$$

Fig. 1. Divide data into subgroups using certain variable index

As an important criterion to assess the accuracy of machine learning predictions, this study determines evaluating error with backtracking.

After that I am using an XGBoost model which creates strong learners based on weak learners. Adding routine models. Therefore, the error of the weak model is corrected by the next model of the chain to obtain the optimal solution as shown in figure below:

```
from xgboost import XGBClassifier

model = XGBClassifier(random_state=1, learning_rate=.1, n_estimators=200)
predictions = backtest(btc, model, predictors)

predictions
```

Fig. 2. Use of XGBoost model for improving precision score

After this I am using various trends for improving precision. One of the keys to success is model accuracy and performance. Model performance is primarily a technical factor, and for some machine learning and deep learning use cases, there is no point in deploying it unless the model is accurate.

RANDOM FOREST CLASSIFIER

Random forest or random decision forest is a supervised Machine Learning algorithm used for classification, regression and other problems using decision trees. A random forest classifier generates a set of decision trees from a randomly selected part of the training set. This is basically a set of decision trees (DT) from a randomly

selected part of the training set, and then collect noise from different decision trees to determine the final prediction.

Random Forest fits several decision tree classifiers in different database subsamples and uses the average to improve prediction accuracy and control overfitting. The sub-sample size is controlled by the max_samples parameter if bootstrap = True (default), otherwise the entire data set is used to build each tree. After creating the random forest classifier model and fitting the training and testing data in our model I am importing the precision score from sklearn metrices to check the precision of my model.

PRECISION SCORE

Classification models are used in classification problems to predict the target class of data samples. The classification model assumes that each instance belongs to one class or another group. It is important to evaluate the performance of classification models in order to reliably use these models in industry to solve real-world problems. In the machine learning classification model, performance measures are used to evaluate the performance of the machine learning classification model in a certain context. This performance measure includes precision, accuracy, recall, and F1-score. Model performance is very important for machine learning because it helps to understand the strengths and limitations of these models when making predictions in new situations.

The precision score is a useful measure of the success of prediction when the classes are very imbalanced. Mathematically, it represents the ratio of true positive to the sum of true positive and false positive.

Precision Score = TP / (FP + TP)

From the above formula, you could notice that the value of false-positive would impact the precision score. Thus, while building predictive models, you may choose to focus appropriately to build models with lower false positives if a high precision score is important for checking the model accuracy.

XGBOOST MODEL

XGBoost is an open-source Python library that provides a gradient boosting framework. It helps to produce highly efficient, flexible and portable models. When it comes to prediction, XGBoost outperforms other algorithms or machine learning frameworks. This is due to increased accuracy and productivity. Combine multiple models into one model to correct errors in the existing models.

XGBoost is built on top of the gradient booster framework. Gradient boosting is a machine learning technique used for classification, regression, and clustering problems. It optimizes the model when predictions are made. In this method, different models are grouped together to perform the same task.

The basic model is known as weak learning. They work on the principle that a weak student makes bad guesses when alone, but makes the best guesses when in a group.

XGBoost creates strong learners based on weak learners. Adding routine models. Therefore, the error of the weak model is corrected by the next model of the chain to obtain the optimal solution. This is known as an ensemble.

IMPLEMENTATION

1 Preparing data for ML Model

I am using yfinance library for downloading the price data of crypto currency. Yfinance is one of the popular Python modules used to collect data online, and with it we can collect Yahoo's financial data. With the help of the yfinance module, we receive and collect the company's financial data (financial indicators, etc.), as well as marketing data history, using the company's functions.

Downloading the bitcoin price data from 17th April 2014 to present date using yfinance library. The data contains 7 columns: ‘Date’, ‘Open’, ‘High’, ‘Low’, ‘Close’, ‘AdjClose’ and ‘Volume’ with total 3132 rows.

After getting the raw data, we have to clean the data or undergo data pre-processing. We can analyse that the columns ‘Close’ and ‘Adj Close’ have the same data. So, it is better to delete the column “Adj Close” because I don’t need redundant data in my model for future prediction.

Now plotting the graph of closing price of bitcoin from our dataset as shown in the figure below:

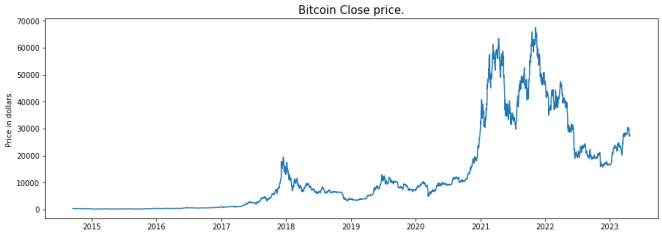

Fig. 3. Variation in the price of bitcoin

Now I am checking for the null values if any present in the DataFrame to proceed further: there are no null values present. So, in order to check for the patterns in the dataset I have created the distribution plot for continuous features from the dataset i.e., ‘close’, ‘open’, ‘high’, ‘low’ as shown in the figure below:

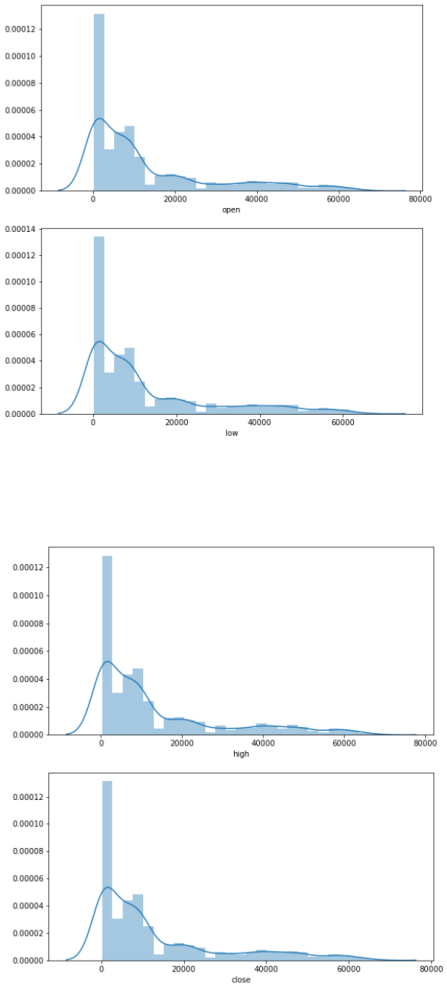

Fig. 4. Distribution Plot for OHLC data

So, after finalizing the dataset I am creating two new important features: ‘tomorrow’ and ‘target’. The tomorrow feature or column will show the closing price for next day i.e., what is the predicted price for a new data in future for trading. The target column will compare the todays closing price and tomorrow closing price to give us the signal whether the market will go up or down in future. This is basically a signal for future trading.

The ‘tomorrow’ and ‘target’ are two very important features in the dataset such that tomorrow tells the closing price of the cryptocurrency for the next day and the target feature implies about the target that is up of low means whether the price of the cryptocurrency will increase or decrease.

I have added the target feature which is a signal whether to buy or not we will train our model to predict this only. But before proceeding let’s check whether the target is balanced or not using a pie chart.

Following is a pie chart between the values of the target variable in the dataset i.e., 0 and 1, where 0 represents the down target means the price of the cryptocurrency will decrease in the future and the 1 value of the target will represent the up value, means the price of the cryptocurrency will go down in the future. I am using the pie chart to represent whether the target variable is balanced or not.

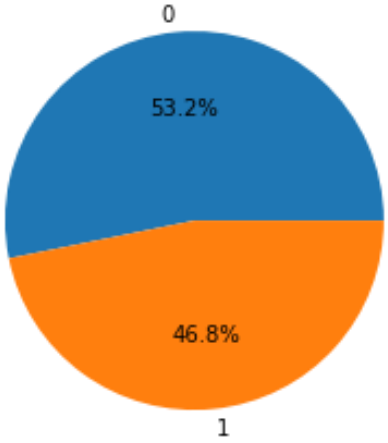

Fig. 5. Pie Chart for target variable

In this pie chat, 0’s represents the negative signal for buying the crypto means the market will fall and the 1’s represent the positive signal which denotes the right time for purchasing the currency.

From this pie chart we can observe that the number of 0’s and the number of 1’s is almost equal or we can say that our data is balanced.

Whenever we add new features to our dataset, we have to ensure that there are no highly correlated features as they do not help in the learning process of the algorithm. For this we will create a heat map to verify or check the correlation between features in our data.

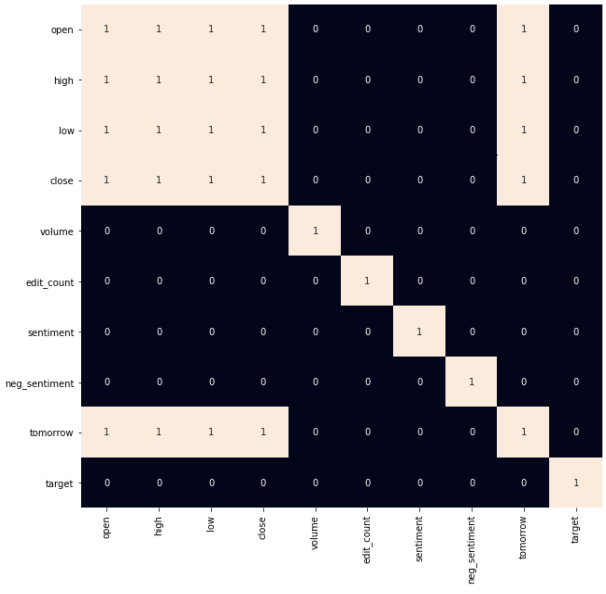

Fig. 6. Heat Map

From the above heatmap, we can say that there is a high correlation between OHLC (open, high, low, close) which is pretty obvious, and the added features are not highly correlated with each other or previously provided features which means that we are good to go and build our model.

2 Training Baseline ML Model

I am using Random Forest classifier to create my model using sklearn. ensemble python library. Scikit-learning is an open-source data analysis library and the gold standard for Machine Learning (ML) in the Python ecosystem. As a high-level library, it allows you to define a predictive data model in a few lines of code, and then fit that model to your data. It integrates well with other Python libraries such as matplotlib for plotting and plotting, numeric for array vectorization, and pandas for data frames.

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(n_estimators=100, min_samples_split=50, random_state=1)

train = btc.iloc[:-200]
test = btc[-200:]

predictors = ["close", "volume", "open", "high", "low", "edit_count", "sentiment", "neg_sentiment"]
model.fit(train[predictors], train["target"])
```

Fig. 7. Using Random Forest Classifier

After creating the random forest classifier model and fitting the training and testing data in our model I am importing the precision score from sklearn metrices to check the precision of my model.

The precision score is a useful measure of the success of prediction when the classes are very imbalanced. Mathematically, it represents the ratio of true positive to the sum of true positive and false positive.

```
from sklearn.metrics import precision_score

preds = model.predict(test[predictors])
preds = pd.Series(preds, index=test.index)
precision_score(test["target"], preds)
```

Fig. 8. Checking Precision Score

From my model I get the precision score of 0.5357142857142857 which is less, the value of precision score lies between 0 and 1. In the forward steps I will work on my model to increase the precision.

3 Evaluating Error with Back Testing

The concept of backtesting is actually quite simple. This is to save the last part of the data as Test data so that the model can be evaluated against the test data. Let's see how it works step by step:

1. Divide the data into Training data and Test data.
2. Build a model based on training data.
3. Use the model to predict the duration of the test data.
4. Compare between expected and actual.

```
def predict(train, test, predictors, model):
    model.fit(train[predictors], train["target"])
    preds = model.predict(test[predictors])
    preds = pd.Series(preds, index=test.index, name="predictions")
    combined = pd.concat([test["target"], preds], axis=1)
    return combined
```

I am using two functions for evaluating error with backtesting, one is predict function as shown in figure above and another is backtest function as shown in figure below:

```
def backtest(data, model, predictors, start=1095, step=150):
    all_predictions = []

    for i in range(start, data.shape[0], step):
        train = data.iloc[0:i].copy()
        test = data.iloc[i:(i+step)].copy()
        predictions = predict(train, test, predictors, model)
        all_predictions.append(predictions)

    return pd.concat(all_predictions)
```

Fig. 9. Evaluating Error with Back Testing

4 Using an XGBoost Model

XGBoost is an open-source Python library that provides a gradient boosting framework. It helps to produce highly efficient, flexible and portable models.

When it comes to prediction, XGBoost outperforms other algorithms or machine learning frameworks. This is due to increased accuracy and productivity. Combine multiple models into one model to correct errors in the existing models.

XGBoost is built on top of the gradient booster framework. Gradient boosting is a machine learning technique used for classification, regression, and clustering problems. It optimizes the model when predictions are made. In this method, different models are grouped together to perform the same task.

The basic model is known as weak learning. They work on the principle that a weak student makes bad guesses when alone, but makes the best guesses when in a group.

XGBoost creates strong learners based on weak learners. Adding routine models. Therefore, the error of the weak model is corrected by the next model of the chain to obtain the optimal solution. This is known as an ensemble.

```
from xgboost import XGBClassifier

model = XGBClassifier(random_state=1, learning_rate=.1, n_estimators=200)
predictions = backtest(btc, model, predictors)

predictions
```

Fig. 10. Using XGBoost Model

After using XGBoost classifier with 200 number of estimators and then using our back testing functions, I am checking the prediction as shown below in the image:

	target	predictions
2017-09-16	0	0
2017-09-17	1	0
2017-09-18	0	1
2017-09-19	0	0
2017-09-20	0	0
...	...	...
2023-04-21	1	0
2023-04-22	0	0
2023-04-23	0	0
2023-04-24	1	0
2023-04-26	0	0

2048 rows × 2 columns

Fig. 11. Predictions after using XGBoost Model

5 Improving Precision with Trends

Machine Learning and Deep Learning models are around us in modern organizations. The number of AI use cases is growing rapidly with the rapid development of new algorithms, cheaper computing, and more data. Banking, healthcare, education, manufacturing, construction, etc., every industry has relevant machine learning and deep learning applications. One of the biggest challenges in all these ML and DL projects across industries is model optimization.

There is no one-size-fits-all strategy to improve existing machine learning and deep learning models. I will review a set of guidelines and best practices that can be evaluated to consistently identify potential sources of improvement in model accuracy and performance.

One of the keys to success is model accuracy and performance. Model performance is primarily a technical factor, and for some machine learning and deep learning use cases, there is no point in deploying it unless the model is accurate.

```python
def compute_rolling(btc):
    horizons = [2, 7, 60, 365]
    new_predictors = ["close", "sentiment", "neg_sentiment"]

    for horizon in horizons:
        rolling_averages = btc.rolling(horizon, min_periods=1).mean()

        ratio_column = f"close_ratio_{horizon}"
        btc[ratio_column] = btc["close"] / rolling_averages["close"]

        edit_column = f"edit_{horizon}"
        btc[edit_column] = rolling_averages["edit_count"]

        rolling = btc.rolling(horizon, closed="left", min_periods=1).mean()
        trend_column = f"trend_{horizon}"
        btc[trend_column] = rolling["target"]

        new_predictors += [ratio_column, trend_column, edit_column]
    return btc, new_predictors
```

Fig. 12. Improving Precision with Trends

	open	high	low	close	volume	edit_count	sentiment	neg_sentiment	tomorrow	target
2014-09-17	465.864014	468.174011	452.421997	457.334015	21056800	0.533333	-0.109741	0.154444	424.440002	0
2014-09-18	456.859985	456.859985	413.104004	424.440002	34483200	0.566667	-0.142785	0.187778	394.795990	0
2014-09-19	424.102997	427.834991	384.532013	394.795990	37919700	0.600000	-0.176097	0.221111	408.903992	1
2014-09-20	394.673004	423.295990	389.882996	408.903992	36863600	0.600000	-0.176097	0.221111	398.821014	0
2014-09-21	408.084991	412.425995	393.181000	398.821014	26580100	0.600000	-0.109894	0.187778	402.152008	1
...	...	...	...	...	...	...	...	...	...	...
2023-04-21	28249.230469	28349.968750	27177.365234	27276.910156	20759504330	0.700000	-0.117651	0.179167	27817.500000	1
2023-04-22	27265.894531	27872.142578	27169.570312	27817.500000	13125734602	0.600000	-0.084758	0.145833	27591.384766	0
2023-04-23	27816.144531	27820.244141	27400.314453	27591.384766	12785446832	0.600000	-0.084758	0.145833	27525.339844	0
2023-04-24	27591.730469	27979.982422	27070.849609	27525.339844	11703288330	0.600000	-0.084758	0.145833	28349.996094	1
2023-04-26	28281.115234	28446.505859	28262.826172	28349.996094	18316503040	0.600000	-0.084758	0.145833	NaN	0

trend_2	close_ratio_7	edit_7	trend_7	close_ratio_60	edit_60	trend_60	close_ratio_365	edit_365	trend_365
NaN	1.000000	0.533333	NaN	1.000000	0.533333	NaN	1.000000	0.533333	NaN
0.0	0.962696	0.550000	0.000000	0.962696	0.550000	0.000000	0.962696	0.550000	0.000000
0.0	0.927789	0.566667	0.000000	0.927789	0.566667	0.000000	0.927789	0.566667	0.000000
0.5	0.970419	0.575000	0.333333	0.970419	0.575000	0.333333	0.970419	0.575000	0.333333
0.5	0.956729	0.580000	0.250000	0.956729	0.580000	0.250000	0.956729	0.580000	0.250000
...	...	...	...	...	...	...	...	...	...
0.0	0.932216	0.666667	0.142857	1.038110	1.554444	0.433333	1.191986	1.514247	0.465753
0.5	0.962443	0.661905	0.285714	1.056418	1.545556	0.450000	1.217347	1.514247	0.468493
0.5	0.967648	0.652381	0.285714	1.045579	1.536667	0.450000	1.209176	1.514155	0.468493
0.0	0.974706	0.642857	0.285714	1.040725	1.527222	0.450000	1.208014	1.514064	0.468493
0.5	1.014416	0.628571	0.285714	1.068436	1.517778	0.466667	1.246020	1.513973	0.468493

Fig. 13. Various Trends in Dataset

As shown in the above table I added a lot of new trends in my dataset which helps my model to predict accurate results and give me good precision scores. This makes my model more accurate and much useful and trustworthy for trading various crypto currencies.

RESULTS AND DISCUSSIONS

Generating Future Predictions

Finally, after making various trends in my dataset as shown in previous section and then again backtesting the model to remove errors using below code:

predictions = backtest (btc, model, new_predictors)

generating the future predictions, model will predict the price accurately and give us the signal whether the price will go up or down using the target variable which I created in the dataset previously in 'Preparing Data for ML Model Section'. The predictions are shown below in the table:

	target	predictions
2017-09-16	0	1
2017-09-17	1	0
2017-09-18	0	0
2017-09-19	0	1
2017-09-20	0	1
...	...	...
2023-04-21	1	1
2023-04-22	0	0
2023-04-23	0	1
2023-04-24	1	1
2023-04-26	0	1

2048 rows × 2 columns

Fig. 14. Final Predictions

Now lets check the precision score between the target and the predictions varible using the below code:

precision_score(predictions["target"],predictions["predictions"])

I got the precision score of:

0.5211406096361848

CONCLUSION

In general, price-related variables are difficult to predict because of the many forces affecting the market. Furthermore, prices are largely based on the future rather than historical data. However, using deep neural networks has given us a better understanding of Bitcoin and LSTM architecture. Ongoing work includes performing hyperparameter tuning to obtain a more accurate network architecture. Also, other features can be considered (more features do not always lead to good results from our tests with Bitcoin). Microeconomic factors can be incorporated into the model for better forecasting results. However, maybe 6 Conclusion In general, given the large number of forces affecting the market, it is difficult to predict the variables related to the price. Furthermore, prices are largely based on the future rather than historical data. However, using deep neural networks has given us a better understanding of Bitcoin and LSTM architecture. Ongoing work includes performing hyperparameter tuning to obtain a more accurate network architecture. Also, other features can be considered (more features do not always lead to good results from our tests with Bitcoin). Microeconomic factors can be incorporated into the model for better forecasting results. However, perhaps the data we have collected for Bitcoin, although collected over the years, can be interesting, making historical interpretations over the past few years. In addition, the successful evolution of peer-to-peer transactions continues and changes the face of payment services. Apparently, with all doubts resolved, the time to act may be perfect. We think it's hard to think Bitcoin is perfect for the future.

REFERENCES

[1] Garcia, D. & Schweitzer, F. (2015) Social signals and algorithmic trading of Bitcoin. Royal Society Open Science 2(9), 150288. Retrieved from https://dx.doi.org/10.1098/rsos.150288 10.1098/rsos.150288

[2] Kathyayini, R. S., Jyothi, D. G. & Crypt". Currency Price Prediction using Machine Learning". *International Journal of Advanced Research in Computer and Communication Engineering*

[3] Alessandretti L, ElBahrawy A, Aiello LM, Baronchelli A (2019) Anticipating cryptocurrency prices using machine learning.

[4] Balcilar M, Bouri E, Gupta R, Roubaud D (2017) Can volume predict Bitcoin returns and volatility? A quantiles-based approach. Econ Model.

[5] Cheah ET, Fry J (2015) Speculative bubbles in Bitcoin markets? An empirical investigation into the fundamental value of Bitcoin. Econ Lett.

[6] Dorfleitner G, Lung C (2018) Cryptocurrencies from the perspective of euro investors: a re-examination of diversification benefits and a new day-of-the-week effect. J Asset Manag.

[7] Foley S, Karlsen JR, Putniņš TJ (2019) Sex, drugs, and bitcoin: how much illegal activity is financed through cryptocurrencies? Rev Financ Stud.

[8] Hyun S, Lee J, Kim JM, Jun C (2019) What coins lead in the cryptocurrency market: using Copula and neural networks models. J Risk Financ Manag.

[9] Jiang Z, Liang J (2017) Cryptocurrency portfolio management with deep reinforcement learning. In: 2017 intelligent systems conference (intelliSys). IEEE, New York.

[10] Mallqui DC, Fernandes RA (2019) Predicting the direction, maximum, minimum and closing prices of

daily Bitcoin exchange rate using machine learning techniques. Appl Soft Comput.

[11] Corbet S, Meegan A, Larkin C, Lucey B, Yarovaya L (2018b) Exploring the dynamic relationships between cryptocurrencies and other financial assets. Econ Lett.

[12] de Souza MJS, Almudhaf FW, Henrique BM, Negredo ABS, Ramos DGF, Sobreiro VA, Kimura H (2019) Can artificial intelligence enhance the Bitcoin bonanza. J Finance Data Sci.

[13] Fang F, Ventrea C, Basios M, Kong H, Kanthan L, Martinez-Rego D, Wub F, Li L (2020) Cryptocurrency trading: a comprehensive survey.

[14] Hyun S, Lee J, Kim JM, Jun C (2019) What coins lead in the cryptocurrency market: using Copula and neural networks models. J Risk Financ Manag.

[15] Erdas, Mehmet Levent, and Abdullah Emre Caglar. 2018. Analysis of the relationships between Bitcoin and exchange rate, commodities and global indexes by asymmetric causality test. Eastern Journal of European Studies.

[16] Guarino, Alfonso, Luca Grilli, Domenico Santoro, Francesco Messina, and Rocco Zaccagnino. 2022. To learn or not to learn? Evaluating autonomous, adaptive, automated traders in cryptocurrencies financial bubbles. Neural Comput & Applic.

[17] Kim, Alisa, Y. Yang, Stefan Lessmann, Tiejun Ma, M.-C. Sung, and Johnnie E. V. Johnson. 2020a. Can deep learning predict risky retail investors? A case study in financial risk behavior forecasting. European Journal of Operational Research.

[18] McNally, Sean, Jason Roche, and Simon Caton. 2018. Predicting the Price of Bitcoin Using Machine Learning. Paper presented at 26th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), Cambridge, UK.

[19] Atsalakis GS, Atsalaki IG, Pasiouras F, Zopounidis C (2019) Bitcoin price forecasting with neuro-fuzzy techniques. Eur J Oper Res.

[20] ação P, Duarte AP, Sebastião H, Redzepagic S (2018) Information transmission between cryptocurrencies: does bitcoin rule the cryptocurrency world. Sci Ann Econ Bus.

[21] Ben-Hur A, Weston J (2010) A user's guide to support vector machines. In: Data mining techniques for the life sciences. Humana Press, London.

[22] Kim YB, Kim JG, Kim W, Im JH, Kim TH, Kang SJ, Kim CH (2016) Predicting fluctuations in cryptocurrency transactions based on user comments and replies.

[23] Kou G, Lu Y, Peng Y, Shi Y (2012) Evaluation of classification algorithms using MCDM and rank correlation. Int J Inf Technol Decis Mak.

[24] Kristoufek L (2013) BitCoin meets Google trends and wikipedia: quantifying the relationship between phenomena of the Internet era.

[25] Lahmiri S, Bekiros S (2019) Cryptocurrency forecasting with deep learning chaotic neural networks. Chaos Solitons Fractals.

[26] Li X, Wang CA (2017) The technology and economic determinants of cryptocurrency exchange rates: the case of Bitcoin. Decis Support Syst.

[27] Panagiotidis T, Stengos T, Vravosinos O (2018) On the determinants of bitcoin returns: a LASSO approach. Finance Res.

[28] Panagiotidis T, Stengos T, Vravosinos O (2019) The effects of markets, uncertainty and search intensity on bitcoin returns. Int Rev Financ Anal.

[29] Patel J, Shah S, Thakkar P, Kotecha K (2015) Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. Expert Syst Appl.

[30] Phillips R C, Gorse D (2017) Predicting cryptocurrency price bubbles using social media data and epidemic modelling. In: 2017 IEEE symposium series on computational intelligence.