Prediction of Bitcoin Price Direction Based on Market Momentum via Logistic Regression

Hesham T. Banafa

Student at the Department of Electrical and Computer Engineering, Faculty of Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia

hbanafa0002@stu.kau.edu.sa
hishaminv@gmail.com

Abstract – In this work, we showcase the ability to predict bitcoin price direction on a daily basis. At the core, the dataset is tailored to be agnostic to absolute price value, but rather the change in price value relative to previous n days, allowing to develop a more general classifier. Binary logistic regression is leveraged to develop a classifier for price direction, up or down for next day. The classifier is evaluated using a standard confusion matrix in combination with Accuracy, Precision, Recall, and F1 score. In this, we produced, consistently, binary direction classifiers with Accuracy of 54%, Precision of 63%. A trading scenario is developed to showcase the performance. A return of 108% of principal is gained, keeping in mind the general upward trend of Bitcoin is factored in.

Keywords – Price prediction · Bitcoin · Cryptocurrency · Machine Learning

Introduction

Ever since the invention of Bitcoin and Blockchain technology, financial institutions disregarded the validity and legitimacy of the technology. However, over time, cryptocurrencies have proven most of the claims of being truly distributed trustless system and database of transactions. The tokens, or coins, themselves have become a valued commodity that is traded in more than 16000 markets, mostly through the internet. Most traders question how the tokens retain value and how it is determined in the first place. Similar to fiat currencies, the value is a reflection of the confidence in the instrument. A common goal for engineers, scientists, and traders is to computationally find correlations based on features to predict successfully at a good rate, the price or direction to gain monetary profits [1] [2]. In previous works, many different machine learning algorithms have been implemented, such as recurrent neural network, long short-term memory, support vector machines. In this paper, we leverage logistic regression and a dataset courtesy of CoinGecko[1], as a more simple approach combined with ensuring that we eliminate unbounded features. For example, the price of tokens is by nature unbounded. Meaning there is no objective reason it could not be 50× in a short-term. Moreover, having unbounded features may lead to overfitting and performance only applies for the range of the features and time frame used during training. Thus, we engineer the features to be relative to $T - n$, where n is the number of days. We hypothesis this approach can be more general and applies better over wider range of time and feature size.

In the scope of this experiment, we do not provide trading strategy or methodology. However, the goal is to be able to produce an efficient model to predict, with accuracy above 50%, the direction of BTC/USD price for $T + 1$, where T is current time, and a set of features at $T + 0$. Moreover, the nature of the present dataset is daily snapshots of price and total trading volume.

Methodology and Dataset

Dataset

The gathered dataset provides a snapshot of the market with the features price, market capitalization, and total trading volume, all in USD. The snapshots are taken at the same time every day at 00:00:00 UTC, or 12AM UTC, or 3AM UTC+3 for Saudi Arabia. The dataset, after clearing incomplete samples, contains 2912 samples, each sample represents a day. The time frame of the data is from 2013-12-30 up to 2021-12-21.

[1]https://www.coingecko.com/en/coins/bitcoin/historical_data/usd

Features

The features, as mentioned before, are limited to the time and magnitude of the the capturing time. Thus, we construct new features from the provided ones. This process in essence opens room for experimenting with different combinations of features and the depth, which we discuss in the following. We construct the new dataset by taking the values of *price* at time $T + 0$, and calculating the percent change relative to $T - 1$, $T - 2$, ... $T - n$ and so on. The same is applied for *total_volume*. Hence, here we have a dataset that represents features of n depth of days, or in other words momentum aware. Note, the dataset is relatively trivial when compared to Chen's [2], where price of gold, blockchain load, transaction fees, and most interestingly written news sources with keyword "Bitcoin". The explicit unawareness of news of regulation, advancements of the Bitcoin technology, is a notable limitation in this experiment and is outside the scope of this paper.

Table 1: Daily features with n=2

Feature	Definition
price_one_day_delta	Percent change in price from T - 1
price_two_day_delta	Percent change in price from T - 2
volume_one_day_delta	Percent change in volume from T - 1
volume_two_day_delta	Percent change in volume from T - 2

Formulation

Logistic Regression (LR) method is applied on the dataset to fit the model for prediction. Given a matrix of training samples

$$X=[x_t^v] \qquad (1)$$

Where t denotes index of time, and v denotes index of feature, the vector

$$Y=\{y_t\} \qquad (2)$$

Contains the corresponding binary labels, constructed by viewing $T + 1$, and assigning 1 if price changes positively, and 0 if negative change occurred. From (1), $x_t^v \in \mathbb{R}$ denotes the value of feature v at time index t. From (2), $y_t \in \{1,0\}$ denotes the label value at time t. Depending on a set threshold of 0.5, a value $y_t > 0.5$ denotes an increase in price for $t + 1$, and $y_t < 0.5$ denotes a decrease in price. The Sigmoid function for the method of binary logistic regression is as follows [3].

$$h(X)=g(\theta^T X)=\frac{1}{1+e^{-\theta^T X}} \qquad (3)$$

$$0 \leqslant h(X) \leqslant 1$$

$$y_t=\theta_0+\theta_1 x_t^1+\theta_2 x_t^2+\ldots+\theta_n x_t^n \qquad (4)$$

For the cost function, the cross entropy function is leveraged as the minimization goal.

$$J(\theta)=\frac{-1}{n}\sum_{i=1}^{n}[y^{(i)}\ln(h(x^{(i)}))+(1-y^{(i)})\ln(1-h(x^{(i)}))]$$

(5)

Results

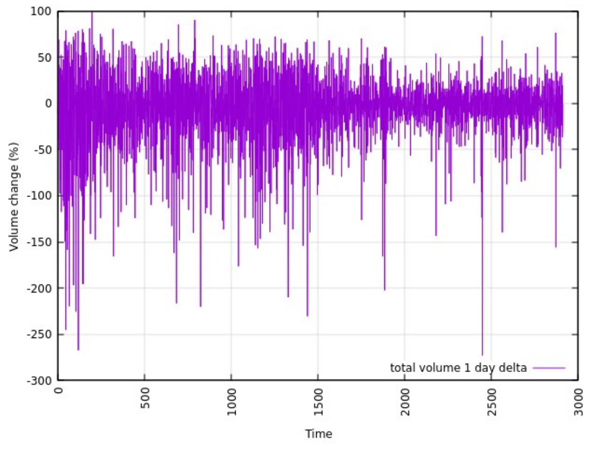

Figure 1: Percent daily change in total trading volume

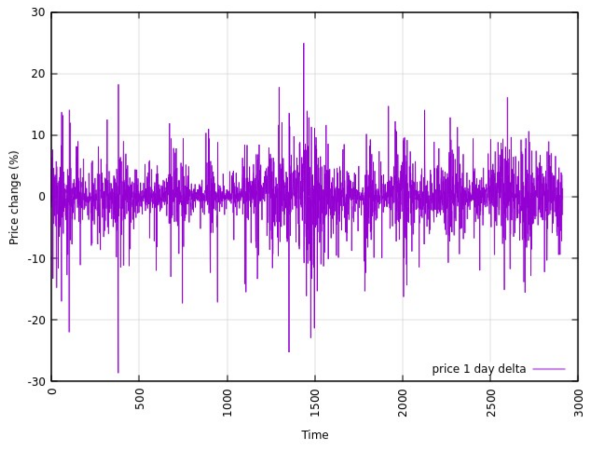

Figure 2: Percent daily change in price

Table 2: Statistics of dataset features

	price_1_day_delta	price_2_day_delta	total_volume_1_day_delta	total_volume_2_day_delta
count		2911		
mean	0.06	0.13	-15.19	-17.19
std	4.02	5.63	456.85	363.39
25%	-1.35	-2.15	-19.87	-25.87
50%	0.18	0.34	-1.75	-1.39
75%	1.8	2.84	14.3	19.64

Table 3: Confusion matrix

Confusion matrix		Prediction	
		1	0
Real	1	True positive(tp)	False negative(fn)
	0	False positive(fp)	True negative(tn)

Evaluation metrics

$$Accuracy=(tp+fn)/(tp+tn+fp+fn) \quad (6)$$

$$Precision=tp/(tp+fp) \quad (7)$$

$$Recall=tp/(tp+fn) \quad (8)$$

$$F1=2\times Precision\times Recall/(Precision+Reca$$

(9)

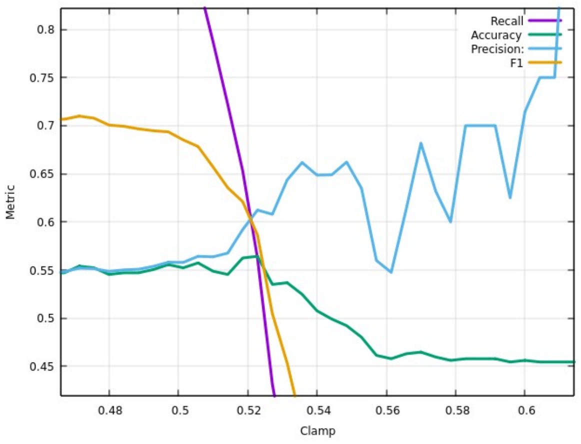

Figure 3: Performance metrics vs clamping threshold

We note that Accuracy alone is not sufficient as a metric for profitability. Certainly a False Negative case would miss a chance for gaining value on a trading position, but does not endure loss of capital. On the other hand, a False Positive case would result in a immanent loss of capital. Depending on a trading strategy, this can be minimized or avoided, however, it is not in the scope of this work. Thus, in this context, we should tune to allow for more False Negative predictions over False Positives. From (7) and (8), we look to optimize Precision over Recall, in combination with an accuracy above 50%.

The following are the performance metrics with clamping at 0.53.

Table 4: Metrics for clamping at 0.5 and 0.53

Metric	Clamping	
	0.53	0.5
Accuracy	0.54	0.55
Precision	0.63	0.56
Recall	0.37	0.9
F1	0.47	0.69

Trading scenario

A trading test is run with the above properties on the testing section of the dataset. A virtual balance of $100 is the initial state. A position is opened if no previous position is open, and the prediction for *t + 1* is 1 or *up*. If a position is open, and the *t + 1* prediction is *up*, the position is kept open. If the prediction is *down*, the position is closed. In an ideal case we would trade at any time of the day. However, here due to the nature of the dataset, we trade only at 12AM UTC.

Table 5: Trading simulation results

Clamp	Peak	End Balance
0.5	392	177
0.53	229	208

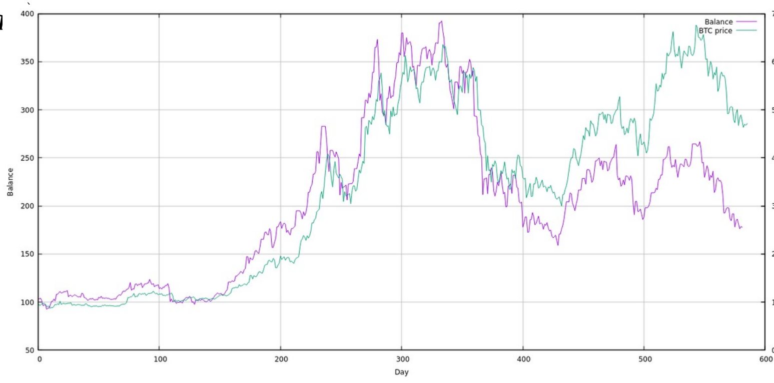

Figure 4: Balance, BTC price vs time (Clamp 0.5)

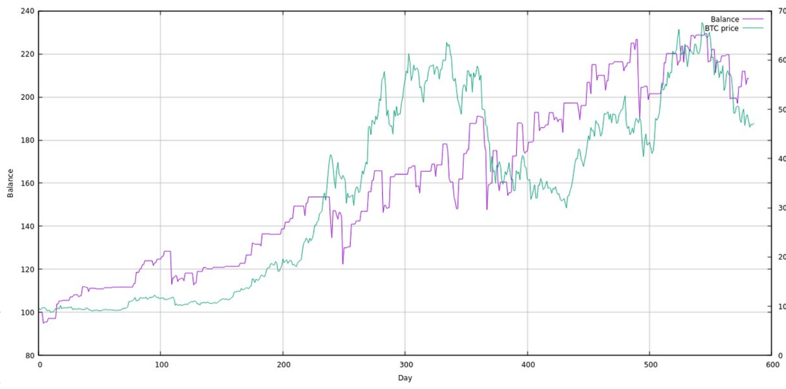

Figure 5: Balanc, BTC price vs time (Clamp 0.53)

From figure 4, it is inferred that the balance of the trading software is an overlay over the actual price of Bitcoin. It is true the end value of the simulation the balance is increased, however, it was the price trend of the market it self. In other words it is similar to buying on day 1, and holding.

From figure 5, the balance is less correlated to the market price. Note from day 360 up to 420, it is inferred market value is trending down, while the balance is increasing, which showcases the desired goal.

Conclusion

In this work, the possibility of applying binary logistic regression to predict the price direction of Bitcoin on a daily bases, based market momentum. The approach we used is relatively simple when compared to other works, with more features allowing deeper analysis. In this work, the price and volume change of *n* previous days are used as features for the logistic regression.

The clamping value is used to optimize the precision, minimizing the false negative case, which inherently causes loss of capital. For a clamping value of 0.53, a precision score of 63% is achieved with an accuracy 54%. The above showcased that a lower precision presents a market mimicking behavior.

Further investigation and experimenting is highly likely to produce more robust classifiers, by testing with more days depth, or different loss function and algorithm. The most interesting field for further research is to use a more fine-grained dataset with, for example, an hourly snapshot. This poses another challenge of finding valid data since there is no one market for trading BTC.

ADDITIONAL INFORMATION

Source code (Case C)

The source code for learning and data manipulation is fully written by myself, except Pandas and numpy and std python libraries.

Model

The model used for testing is named **test2.json**.

Graphs and plots

All plots are done using pandas to aggregate data in combination with GNUplot to plot.

References

1: Isaac Madan,Shaurya Saluja,Aojia Zhao, Automated Bitcoin Trading via Machine Learning Algorithms, 2014

2: Zheshi Chen,Chunhong Li,Wenjun Sun, Bitcoin price prediction using machine learning: An approach, 2019

3: Dr. Muhammad Bilal, Introduction to Artificial Intelligence, Parametric Models – Classification using Logistic, ,