Comparison of Machine Learning Models for Bitcoin Price Prediction

Žan Pečovnik
Faculty of Computer and Information Science, University of Ljubljana, Slovenia
Email: zp8358@student.uni-lj.si

***Abstract*—In the last decade the interest in crypto currencies has greatly increased. Many saw this field of computer science as an investment opportunity while others saw it as much more than just profit. In 2008 Satoshi Nakamoto presented a new digital asset called Bitcoin and just in a few years, this currency became the most popular crypto currency in the world. With its price climbing into the sky people started to use machine learning models to predict the price of Bitcoin. One must take into consideration a lot of different factors when predicting the bitcoin price, from the historical price data, to what is happening in the geopolitical parts of the world and even to social media. If any machine learning algorithm could process all of these things together, it could predict the price very accurately. In this article we are going to keep things more casual and predict the bitcoin price only using the historical price data. We are going to use six different machine learning regression models for prediction and measure the success with three different metrics. The most successful model is the Linear Regression, while the least successful is the Decision Tree Regression.**

***Index Terms*—Linear Regression (LR), Huber Regression (HR), Random Forest Regressions (RFR), Gradient Boosting Regression (GBR), Decision Tree Regression (DTR), Support Vector Machine - Regression (SVM), Bitcoin (BTC)**

I. Introduction

SINCE technology has made an enormous step forward in the past decade, a lot of different mechanisms of payments have emerged to the surface. In 2008 Satoshi Nakamoto invented a crypto currency called Bitcoin. It is a decentralized digital currency without a central bank or a single administrator. Transactions that are made with this currency are verified by network nodes through cryptography and recorded in a public distributed ledger called blockchain which can not be tempered with. Bitcoins are created as a reward for a process called mining. In 2011 one unit of Bitcoin was worth 0,30 USD, but today it is worth approximately 27.000 USD and the estimations are that the price could rise in the next years all the way up to 400.000 USD. A lot of people see this as an opportunity for investment, and alongside this opportunity came a desire for predicting the price of Bitcoin.

In this article we will try to predict the price using six different machine learning regression models and we will try to measure the success of these models by calculating three different metrics and then compare the models to eachother. We will describe these models and metrics in detail down below.

Since there are a lot of factors which can influence the price of bitcoin, such as geopolitical news, current state of the world economy, twitter and facebook feed and many more, studies have shown that the best machine learning models for predicting this type of time series problems are Long Short Term Memory and Recurrent Neural Networks because they can optimize the prediction based on all of these previously mentioned factors. Nevertheless we will not use these models and will only focus on regression models which take into account only the historical data.

A. Used Machine Learning Models

1) Random Forest Regression (RFR): Random Forest Regression is based on combining many Decision Tree Regressions together, where each of these Decision Tree Regression is individually fitted to the training data and then individually tested on the testing data. The output of the Random Forest Regression is an average of all of the individual predictions of every single Decision Tree. Random Decision Tree Forests correct for Decision Tree's

habit of overfitting to the training dataset.

2) Linear Regression (LR): Linear Regression fits a linear model with coefficients in such way that it minimizes the residual sum of squares between the observed targets in the dataset. Linear Regression models are often fitted using the least squares approach, but they may also be fitted by minimizing the "lack of fit" in some other norm (least absolute deviations regression) or by minimizing a penalized version of the least squares cost function (ridge regression, lasso).

3) Huber Regression (HR): Huber Regression is a variant of Linear Regression which is robust to the outliers. It uses two loss functions, squared loss and absolute loss, which the algorithm then optimizes in such way that the loss functions are not heavily influenced by the outliers while not completely ignoring their effect. The parameter epsilon controls the number of samples that should be classified as outliers. The smaller the epsilon, the more robust it is to outliers.

4) Gradient Boosting Regression (GBR): Gradient Boosting Regression builds an additive model in a forward stage-wise fashion. It optimizes arbitrary differentiable loss functions and in each stage fits a regression tree on the negative gradient of the given loss function. The parameter n_estimators represents the number of boosting stages to perform, the parameter learning_rate shrinks the contribution of each tree and the parameter loss represents the loss function to be optimized.

5) Decision Tree Regression (DTR): Decision Tree Regression represents a decision tree which has many paths from the root to the leaves, where each path represents a series of decisions in inner nodes which leads us to the final target value in the leaves. This model is very popular since it is known for its intelligibility and simplicity.

6) Support Vector Machine (SVM): Support Vector Machine or Support Vector Regressions is a model which depends only on a subset of the training dataset, because the cost function for building the model ignores any training data close to the model prediction. The parameter kernel specifies the kernel type to be used in the algorithm, the parameter gamma is a kernel coefficient and the parameter C is used for regularization.

B. Used Metrics

1) Mean Squared Error (MSE): MSE is a measure of the quality of a prediction model. It is always non-negative, and values closer to zero are better. It is described by the equation below.

$$MSE = \frac{1}{n}\sum_{1}^{n}(y_i - \hat{y}_i)^2 \tag{1}$$

2) Mean Absolute Error (MAE): MAE is the sum of absolute differences between our target and predicted variables. It measures the average magnitude of errors in a set of predictions, without considering their directions. It is always non-negative and values closer to zero are better. It is described by the equation below.

$$MAE = \frac{1}{n}\sum_{1}^{n}|y_i - y| \tag{2}$$

3) R Squared (R^2): R^2 is a statistical measure of how well the regression predictions approximate the real data points in other words it gives some information about the goodness of the fit of a model. It ranges between 0 and 1, and the closer the value is to 1, the better is the fit of the model. If the value is 1, this means that the model fits perfectly to the data. It is described by the equation below.

$$R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2} \tag{3}$$

II. Related Work

For better understanding of the underlying problem of predicting bitcoin price we reviewed a few articles.

The first article, written by Thearasak Phaladisailoed and Thanisa Numnonda [1] represents different models and how to use them for predicting the price, it was especially helpful for us, because we also use one of the described models Huber regression (HR). The article also gave us an additional insight about the problem of predicting bitcoin price based on a time series.

The second article, written by Karunya Rathan, Somarouthu Venkat Sai and Tubati Sai Manikanta

[2] represents two different machine learning models both of which we used in our research, Decision Tree regression (DTR) and Linear regression (LR). Both of previously mentioned models are described in detail, step-by-step.

The third article, written by Amin Azari [3] represents a different model called Autoregressive Integrative Moving Average (ARIMA) which we did not use in our research, but the whole process of predicting, the preprocessing of the data and even performance of the algorithm was very well described and helped us a lot.

The fourth article, written by Poongodi M., Ashutosh Sharma, Vijayakumar V., Vaibhav Bhardwaj, Abhinav Parkash Sharma, Razi Iqbal and Rajiv Kumar [5] represents two machine learning models Linear regression (LR) and Support Vector Machine (SVM) which we also used in our research. In the mentioned article they used these two methods for predicting the price of Ethereum, but the concept is very simmilar to predicting the price of Bitcoin, since in both cases it is a time series prediction.

III. Data

For predicting future Bitcoin prices we used the historical data found on Kaggle website [17]. The Bitcoin price data covers the data from 18th August 2017 to 11th February 2020 with hourly frequency and currency of USD. Dataset contains six different columns.

- **date** - date and time
- **open** - bitcoin price at the start of the hour
- **high** - highest bitcoin price in the hour
- **low** - lowest bitcoin price in the hour
- **close** - bitcoin price at the end of the hour
- **vol** - trading volume in the hour

	date	open	high	low	close	vol
0	2017-08-18 00:00:00	4285.08	4340.62	4228.76	4286.53	82.435474
1	2017-08-18 01:00:00	4269.36	4269.36	4134.61	4243.59	75.684883
2	2017-08-18 02:00:00	4251.95	4298.64	4234.00	4267.59	54.071337
3	2017-08-18 03:00:00	4244.77	4308.70	4234.00	4292.39	64.001349
4	2017-08-18 04:00:00	4292.39	4292.39	4234.43	4287.92	50.797515
5	2017-08-18 05:00:00	4287.92	4340.62	4250.04	4313.56	64.169286
6	2017-08-18 06:00:00	4313.56	4316.98	4247.75	4279.46	64.235424
7	2017-08-18 07:00:00	4279.46	4304.67	4270.48	4300.25	69.186804
8	2017-08-18 08:00:00	4285.00	4320.71	4259.85	4282.73	74.829849
9	2017-08-18 09:00:00	4265.33	4327.25	4265.33	4304.15	45.653156

Fig. 1. Bitcoin price historical data

Before using any machine learning process on our dataset, we have to prepare the data. We excluded all attributes except the opening price in the hour and created a new attribute called prediction. This column contains prices of bitcoin after a 48 hour timespan.

	price	prediction
0	4285.08	4139.98
1	4269.36	4086.09
2	4251.95	4082.53
3	4244.77	4124.69
4	4292.39	4094.62
5	4287.92	4093.00
6	4313.56	4117.41
7	4279.46	4115.41
8	4285.00	4155.87
9	4265.33	4184.73

Fig. 2. Bitcoin price and bitcoin price after 48 hours

After the preprocessing was done, we ended up with a dataset of 21620 examples which we then divided into a training (80% = 17296 examples) and testing (20% = 4324 examples) set.

IV. Results

In this chapter we will further describe the predictions of Bitcoin price for different machine learning

models. Below in Figure 3 is an example of how the source code is written for the models, eventhough we have six different models, the code is very simmilar for each one of them. The first thing to do was to fit the regression model to the training dataset. After this was done, we had to make some predictions on the testing dataset so we could then calculate the previously described metrics. We saved the results of metrics calculation so we could plot them later.

```
# Ustvarimo in natreniramo Linearno Regresijo
linear = LinearRegression()
linear.fit(x_train, y_train)

# Testiramo naučen model
linear_predictions = linear.predict(x_test)
linear_mse = mean_squared_error(y_test, linear_predictions)
linear_mae = mean_absolute_error(y_test, linear_predictions)
linear_r2 = r2_score(y_test, linear_predictions)

mses.append(linear_mse)
maes.append(linear_mae)
r2s.append(linear_r2)

# Izpis metrik
print("Linear mean squared error: ", linear_mse)
print("Linear mean absolute error: ", linear_mae)
print("Linear r2 score: ", linear_r2)
```

```
Linear mean squared error:  152755.1263543451
Linear mean absolute error:  264.5058775699
Linear r2 score:  0.8769929501483997
```

Fig. 3. Example of source code for linear regression

The first used machine learning model is the Random Forest Regression with the number of estimators equal to 50, which means that 50 decision trees are fitted on the training dataset and are then averaged to improve accuracy and control over-fitting. We can see in Figure 4 how the model made the predictions in comparison to the actual price.

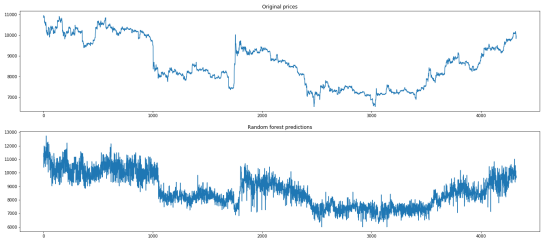

Fig. 4. Comparison of predictions of Random Forest Regression and actual prices

The second used machine learning model is the Huber Regression with the parameter epsilon equal to 1.35 which controls the number of samples that should be classified as outliers. This is very simmilar to the Linear Regression model but is more robust to the outliers. We can see in Figure 5 how the model made the predictions in comparison to the actual price.

Fig. 5. Comparison of predictions of Huber Regression and actual prices

The third used machine learning model is the Linear Regression which fits the model with coefficients to minimize the residual sum of squares between the observed targets. We can see in Figure 6 how the model made the predictions in comparison to the actual price.

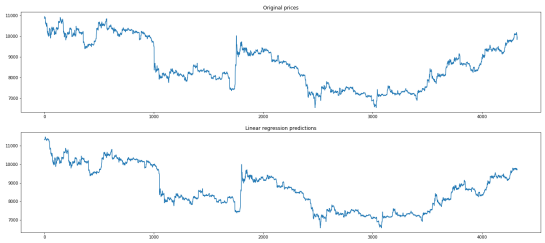

Fig. 6. Comparison of predictions of Linear Regression and actual prices

The fourth used machine learning model is the Gradient Boosting Regression with parameters n_estimators=100, learning_rate=0.1 and loss=ls, where ls refers to least squares regression (the loss function to be optimized). We can see in Figure 7 how the model made the predictions in comparison to the actual price.

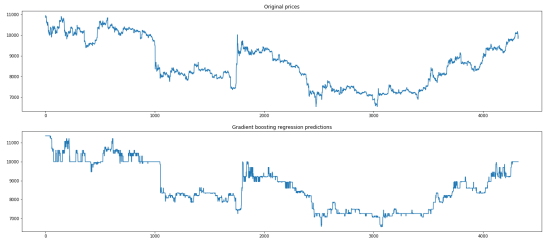

Fig. 7. Comparison of predictions of Gradient Boosting Regression and actual prices

The fifth used machine learning model is the Decision Tree Regression which predicts the value of a target variable by learning simple decision rules inferred from the data features. We can see in Figure 8 how the model made the predictions in comparison to the actual price.

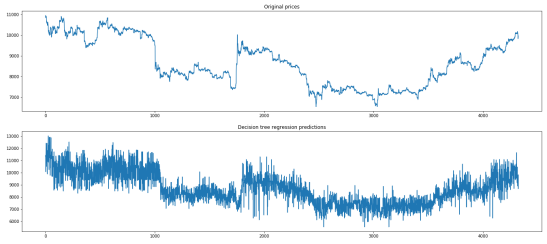

Fig. 8. Comparison of predictions of Decision Tree Regression and actual prices

The sixth and last used machine learning model is the Support Vector Machine, more accurately Epsilon-Support Vector Regression.with parameters kernel='rbf', gamma=0.00002 and C=1e3. We can see in Figure 9 how the model made the predictions in comparison to the actual price.

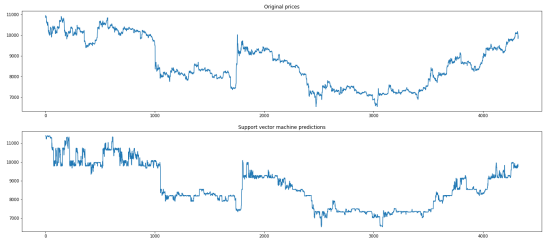

Fig. 9. Comparison of predictions of Support Vector Regression and actual prices

In Figure 10 we can see what the actual price is and then all of the predictions next to it, so we can easily compare which of the models made the best prediction. We can ascertain from the data that in most cases, the best model predicts the price which differs from the original price for about 300 USD, while the worst prediction is almost 1000 USD higher or smaller than the original price.

	Actual price	RFR prediction	HR prediction	LR prediction	GBR prediction	DTR prediction	SVM prediction
0	10952.74	11199.67600	11395.898251	11315.789837	11351.031734	11351.63	11393.092121
1	10852.21	11450.05420	11419.434163	11338.820709	11351.031734	11427.68	11385.074995
2	10908.27	11287.02400	11388.577081	11308.625766	11351.031734	10767.60	11392.606047
3	10925.57	10404.40860	11414.426522	11333.920523	11351.031734	10072.58	11388.013018
4	10936.28	11243.05360	11402.838841	11322.581494	11351.031734	11081.84	11392.260834
5	10914.24	11991.22140	11452.474577	11371.152132	11351.031734	12428.83	11350.649187
6	10896.06	11304.64360	11521.940570	11439.127505	11351.031734	11288.13	11244.672915
7	10900.95	10678.36664	11531.415026	11448.398656	11351.031734	10676.04	11234.203614
8	10893.36	10698.87900	11556.713627	11473.154393	11351.031734	10509.99	11218.679954
9	10761.07	10865.89800	11498.344566	11416.037831	11351.031734	10820.32	11278.289277
10	10617.54	11277.34300	11512.045471	11429.444739	11351.031734	10874.95	11257.692593
11	10690.23	11529.76160	11464.943602	11383.353594	11351.031734	11397.54	11332.331005
12	10654.92	11790.05520	11464.492915	11382.912578	11351.031734	11893.86	11333.027118
13	10571.97	11926.32800	11453.426029	11372.083168	11351.031734	12428.83	11349.329908
14	10623.22	11588.97240	11416.409548	11335.860997	11351.031734	11394.06	11386.928221

Fig. 10. Actual Bitcoin price and all predictions

V. Conclusion

In this article we wanted to make predictions of Bitcoin price with six different machine learning models. We discovered that Linear Regression and Huber Regression are the best at forecasting the price and both had the R^2 score just above 0.87, while the Random Forest Regression and Decision Tree Regression are much worse and had the R^2 score around 0.57 for the case of Decision Tree Regression and around 0.72 for the case of Random Forest Regression. All of the calculated metrics can be seen in Figure 11 for easier comparison.

	MSE	MAE	R2
Random forest regression	347458.182453	443.990929	0.720207
Huber regression	157729.268518	269.234059	0.872987
Linear regression	152755.126354	264.505878	0.876993
Gradient boosting regression	171574.524951	285.066068	0.861839
Decision tree regression	530758.203452	553.113380	0.572604
Support vector machine	180600.692391	295.877680	0.854570

Fig. 11. MSE, MAE and R^2 score for all models

Therefore we can conclude that the best model in our case is the Linear Regression since it has the highest R^2 score and the lowest MSE and MAE of all the models. But predicting Bitcoin price is not just as simple as looking at some historical data, we should also take into consideration a lot of other factors, which play a very important role and affect the price a lot. Some of these other factors are daily news from the political, social and economical part of the world. If we would really like to make an accurate prediction, we should also include these factors alongside with the historical data.

REFERENCES

[1] Phaladisailoed, Thearasak and Numnonda, Thanisa. Machine Learning Models Comparison for Bitcoin Price Prediction. *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)*

[2] Rathan, Karunya and Venkat Sai, Somarouthu and Sai Manikanta Tubati. Crypto-Currency price prediction using Decision Tree and Regression techniques. *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*

[3] Azari, Amin. Bitcoin Price Prediction: An ARIMA Approach. https://arxiv.org/pdf/1904.05315.pdf

[4] Keerthi, Mukul. https://medium.com/@mukul04.sk/how-to-choose-the-best-regression-model-for-your-ml-application-d1e60d6a98ad

[5] Poongodi M. and Ashutosh Sharma and Vijayakumar V. and Vaibhav Bhardwaj and Abhinav Parkash Sharma and Razi Iqbal and Rajiv Kumar. Prediction of the price of Ethereum blockchain cryptocurrency in an industrial finance system. http://www.sciencedirect.com/science/article/pii/S0045790618331343

[6] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

[7] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.HuberRegressor.html

[8] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[9] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

[10] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html

[11] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

[12] https://www.youtube.com/watch?v=AaEbtnplqjw&ab_channel=ComputerScience

[13] https://sl.wikipedia.org/wiki/Bitcoin

[14] https://en.wikipedia.org/wiki/Mean_squared_error

[15] https://en.wikipedia.org/wiki/Mean_absolute_error

[16] https://en.wikipedia.org/wiki/Coefficient_of_determination

[17] https://www.kaggle.com/quartier13/bitcoin-historical-price-1h2017820202