Bitcoin price prediction

Student: Renáta Kovács
Central European University
Quellenstrasse 51, 1100 Vienna, Austria
e-mail: kovacs_renata@student.ceu.edu

ABSTRACT

This project aims to compare different models in the prediction of the price of Bitcoin in United States Dollars between 2025.08.01. and 2025.11.18. We used two regression models, Random Forest regressor and XGBoost, to see if they can predict the price. Our training dataset contained data from 2020.11.01. until 2025.07.31. We evaluated our models using the mean absolute error and root mean squared error metrics. For both models the root mean squared error was below 2300 and 1900 respectively and the mean absolute error was below 1800 and 1300 respectively.

1 INTRODUCTION

We decided to create a Bitcoin price prediction model based on daily price statistics. We gathered data of the Bitcoin price in United States Dollars (USD) and tried to predict the closing price based on the data provided. The results will be trained models that can predict the closing price of Bitcoin based on the opening, highest and lowest price of bitcoin, its transaction volume and the percentage change between opening and closing values.

2 DATA

We downloaded the dataset from investing.com ("investing.com). The dataset includes entries between 2020.11.01. and 2025.11.18. The data included the closing, opening, highest and lowest price of Bitcoin, the volume traded and the change in percentage between opening and closing prices. The data was collected by the student on 2025.11.19.

2.1 Data description

The dataset contained 1844 observations and 7 attribute. The observations contained daily data. All attributes had the original datatype of object and later needed to be changed to the correct format. We had one nominal and six numerical attributes.

The target variable is the closing price of the day. Closing price is the last traded value. In case of Bitcoin, it is the value at which Bitcoin was traded at 23:59:59 of a given day. The target variable has an uneven distribution shown in Figure 1. There is a higher frequency of prices below 70 000 USD then above it. The attributes of the data are the opening, highest and lowest price, the volume traded in the market and the percentage change in price.

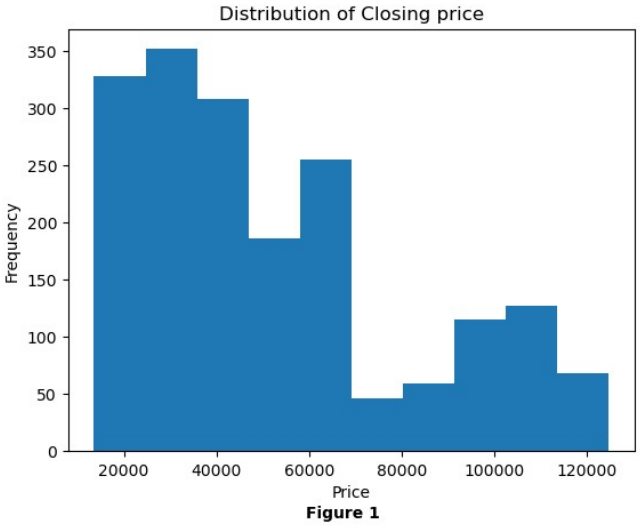

Figure 1

The dataset did not include any missing values. The data had the following format:

	Date	Price	Open	High	Low	Vol.	Change %
0	11/18/2025	93,001.2	92,196.6	93,781.8	89,391.9	115.18K	0.87%
1	11/17/2025	92,195.0	94,259.8	96,002.7	91,327.8	109.89K	-2.19%
2	11/16/2025	94,259.8	95,587.6	96,575.1	93,043.5	68.24K	-1.39%
3	11/15/2025	95,587.6	94,557.5	96,799.5	94,557.5	48.15K	1.09%
4	11/14/2025	94,554.3	99,707.0	99,843.9	94,126.2	131.30K	-5.19%

Table 1 - Data

The visualization for the Closing Price on a given Date is in Figure 2.

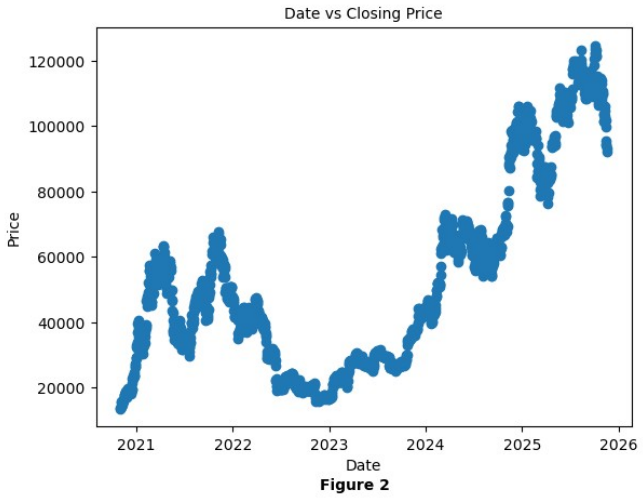

Figure 2

2.2 Data understanding

The dataset had seven attributes. The first attribute was the date, which needed to be transformed into datetime format. It described the date on which the data was collected. It is a unique value between 2020.11.01 and 2025.11.18. The next four attributes described the price of Bitcoin during a calendar day. The second attribute was Price, which gave the closing price of Bitcoin for the given calendar day. This is the target variable.

The third attribute was Open, which indicated the opening price of Bitcoin for the given day. It had a maximum value of 124,687.5 USD and a minimum of 13,560 USD. The attribute has a similar distribution to the target variable as seen in the first plot of Figure 3. Opening price is an indication of the target variable.

The fourth attribute was High, which recorded the highest price for which a trade was made in Bitcoin for the given day. It has a maximum value of 126,186 USD and a minimum of 13,828.4 USD. The attribute has a similar distribution to the distribution of the target variable as seen in middle plot of Figure 3.

The fifth attribute was Low, which recorded the lowest price for which a trade was made. The attribute has a maximum value of 123,144.6 USD and a minimum value of 13,214.2 USD. The attribute has a similar distribution to the distribution of the target variable as seen in the last plot of Figure 3.

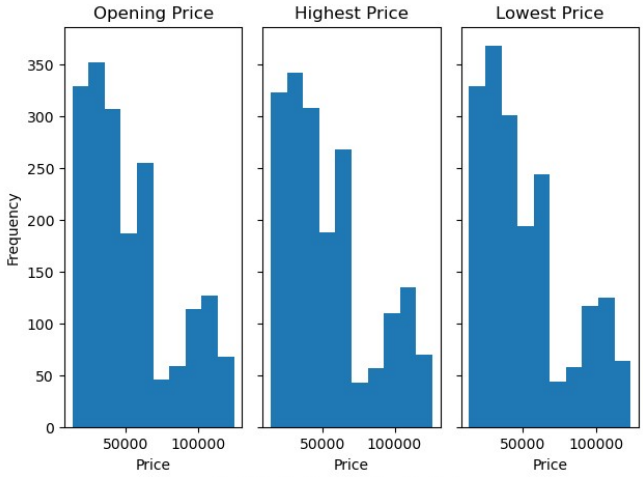

Figure 3 - Distribution of Price indicators

The sixth attribute was the volume traded in the market. Volume is defined as the number of transactions using Bitcoin on a given day. The attribute has an extremely skewed distribution to the left as seen in Figure 4. This is due to the changes in order of magnitude in the data. There are days in the dataset that had billions of transactions and there are also days that only have a couple thousand transactions.

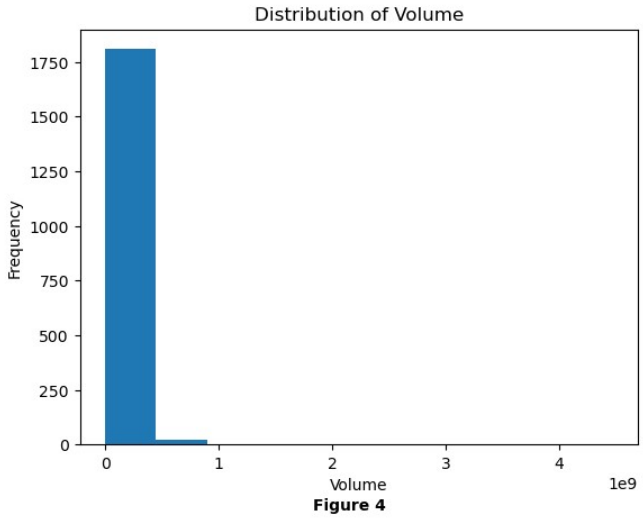

Figure 4

The seventh attribute was the daily percentage change between the opening and the closing price of Bitcoin in the market. The attribute is normally distributed as seen in Figure 5 with a maximum value of 19% and a minimum value of -15%.

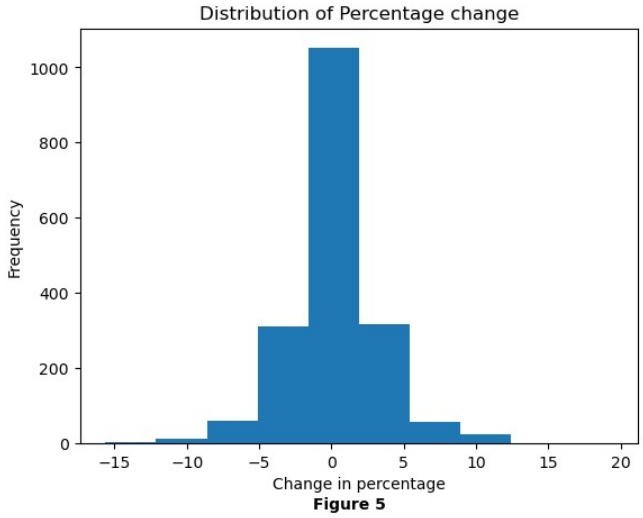

Figure 5

2.3 Data preprocessing

To start preprocessing we created a pipeline that converted the dataset. The first step was to rearrange the dataset. The original dataset included the data in reverse chronological order, therefore we needed to switch the order of the dataset to ensure that the data is chronological. The second step was to clean the values into the correct format. The datatypes of the attributes were also switched to numerical. The third step was feature engineering. We created an additional eight attributes. We created five attributes to describe the date of the transaction better. The first additional attribute was the year, the second one was the month, the third was the day of the transaction. The fourth attribute was the day of the week with Monday being zero and Sunday being six. The fifth attribute was the quarter of the year. The last three attributes were rolling means. We created the three-day, seven-day and fourteen-day rolling mean. We set the index of the dataset to be the Date

attribute. In a separate function we normalized the four attributes regarding the price of Bitcoin: Price, Open, High and Low. We used log-normalization to smooth out high peaks of the data. We separated the dataset based on the date 2025.08.01. into a training and test dataset. We normalized the volume of trades using min-max normalization on the training dataset. The test dataset's volume column was transformed using the scale of the training dataset.

3. MACHINE LEARNING METHODS USED

We chose two methods as possible candidates. The first model was an XGBoost regression model ("XGBoost Parameters"). XGBoost is a gradient boosted decision tree algorithm. The second model was a Random Forest regression model ("RandomForestRegressor"). The Random Forest regression model is a multiple decision tree algorithm. The trees are trained on different subsets of the training data to ensure variability. We chose these models, because these models are good at handling non-linear data. Both models include several smaller models which ensures that the chance of overfitting is low. We used last days closing price as a baseline for the two models.

3.1 Brief description of the methods used

The first method used was the XGBoost regressor. The method builds multiple simple trees based off each other. The second tree learns from the error of the first. The parameters help stabilize the model in size and performance. The most important parameters are learning_rate, the learning rate of the model; eval_metric, the evaluation criteria used; max_depth, the maximum number of levels in a single tree; n_estimators, the number of trees; reg_alpha; L1 regularization term and reg_lambda, L2 regularization term.

The second method used was the Random Forest regressor. The method builds multiple decision trees and trains them using a subset of the training data. It aggregates the vote of the decision trees into a single decision. It shares parameters with the Decision Tree regressor and has additional parameters to control the number and characteristic of the inner trees. The parameter n_estimators sets how many decision trees should be within the forest. Other important parameters are shared with the Decision Tree Regressor model.

3.2 Brief description of the evaluation criteria

We had three evaluation criteria all of them are standard for regression problems. We used the mean squared error and the mean absolute error. To ensure that the units are in dollars we also looked at the root mean squared error. All of the above mentioned three methods evaluate the Euclidean distance in the imagined space between the actual value and the theoretical value. These evaluation criteria are suitable, because they create a measure that shows how close the predictions were to the actual data.

4. EXPERIMENTS

We implemented the Random Forest regressor from the scikit-learn library and XGBoost regressor from the XGBoost library. For hyperparameter tuning we used Optuna. For the splitting of the dataset into training and validation sets we used the Time Series Split class by scikit-learn. ("TimeSeriesSplit") This is a modified version of the K-fold cross validation. Instead of taking random elements it does walk-forward validation, which respects the time dependency of the dataset. Therefore, Optuna performed the hyperparameter tuning using the 5-fold Time Series Split cross validation. With XGBoost the scoring was based on the mean value of the root mean squared error. For the other two models the scoring was based on the mean value of the mean squared error.

Our baseline was the previous day's closing price. Our results showed a mean absolute error of 1692.80 and a root mean squared error of 2194.16.

Our first model was the XGBoost regressor. Hyperparameter tuning was done on the following parameters: n_estimators, the tuned parameter is 3601; max_depth, the tuned estimator is 4; subsample, the tuned estimator is 0.8791118085065366; learning rate, the tuned parameter is 0.1; min_child_weight, the tuned parameter is 15; reg_alpha, the tuned parameter is 0 and reg_lambda, the tuned parameter is 0.5. Our results showed a mean absolute error of 1343.99 and a root mean squared error of 1889.11.

Our second model was the Random Forest regressor. Hyperparameter tuning was done on the following parameters: n_estimators, the tuned parameter is 3236; max_depth, the tuned estimator is 10; min_samples_split, the tuned parameter is 2 and max_features, the tuned parameter is log2. Our results showed a mean absolute error of 1772.79 and a root mean squared error of 2255.00.

Our evaluating criteria are:

Model	MAE	RMSE	MSE
Baseline	1692.80	2194.16	4,814,323
XGBoost	1343.99	1889.11	3,568,740
Random Forest	1772.79	2255.00	5,085,010

Table 2 – Evaluation criteria summary

Compared to our baseline the XGBoost model performs better and the Random Forrest regressor performs worse. This result was a slight surprise, due to the data received by the models. The baseline consists of the previous day's closing price; however, the models did not receive this information. The XGBoost could still perform better than baseline, despite not having this crucial piece of information.

5. VISUALIZATION

The predictions of the models are best represented visually. In all of the Figures below the blue dots represent the factual test data. The other colours are predictions.

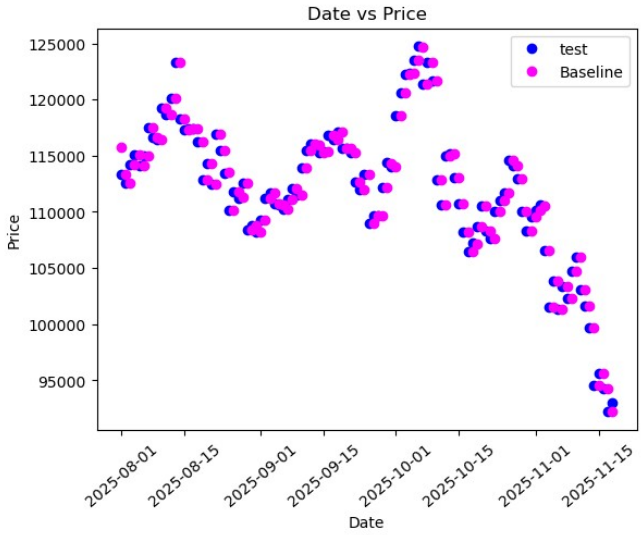

Figure 6 - Baseline

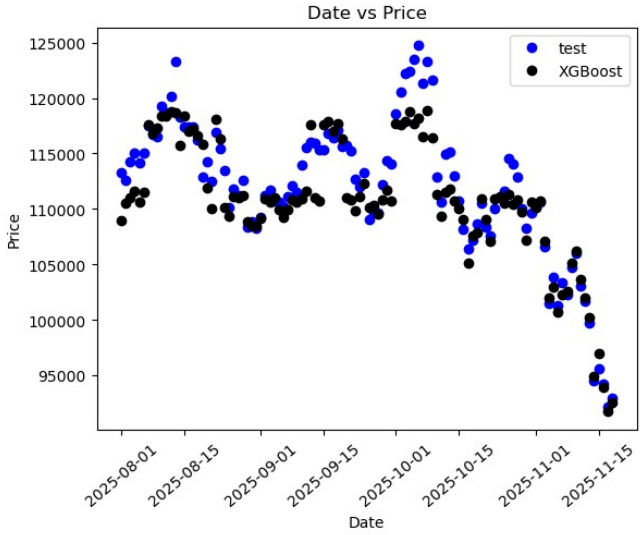

Figure 7 - XGBoost regressor results

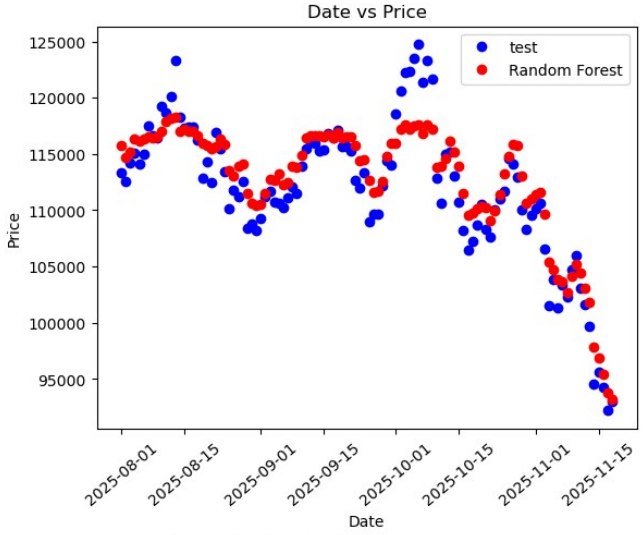

Figure 8 - Random Forest regressor results

For all three estimators, the models cannot predict the peaks in the data. This is due to the rolling means, which dampen the significance of peaks by averaging over a longer period. Based on the evaluation criteria the XGBoost regressor model is better.

The final evaluation criteria are summarized in Table 2.

6. CONCLUSION

In this project we tried to predict the price of Bitcoin between 2025.08.01. and 2025.11.18 using data from 2020.11.01 to 2025.07.31. We used a Random Forest regressor and an XGBoost regressor model. The main outcome of the project is that the XGBoost outperforms the baseline on all evaluation metrics. However, the Random Forrest regressor is worst at predicting the outcome, than the baseline. This project aims to demonstrate predictive improvements compared to a naïve baseline. A possible extension of the project would be to look at other stocks and cryptocurrencies and see if machine learning models are as good at predicting them.

References:

[1]”investing.com”, Accessed: 2025.12.15. https://www.investing.com/crypto/bitcoin/historical-data

[2]”RandomForestRegressor”, Accessed: 2025.12.15. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[3] ”TimeSeriesSplit”, Accessed: 2025.12.15. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html

[4]”XGBoost Parameters” Accessed: 2025.12.15. https://xgboost.readthedocs.io/en/stable/parameter.html