Predicting Tomorrow's Bitcoin Value with Linear and Polynomial Regression

1. Introduction

Cryptocurrency is the cutting-edge transaction mechanism in this era and bitcoin is the most widely used cryptocurrency. However, its value (in US dollars) fluctuates a lot every day. Hence, people who invested in bitcoin can lose significant amounts in the short run. Since most people cannot follow stock exchange frequently, they generally prefer staying away from bitcoin.

In this regard, we aim to make cryptocurrency more controlled and risk minimized for ordinary people like us. Our goal is to approximately predict tomorrow's bitcoin value (closing price in US dollars) and be alarmed before unexpected fluctuations.

This report is composed of 6 main sections. The 1st section introduces the problem and gives an overview about the project content. The 2nd section includes the formulation of the problem in the context of machine learning. The 3rd section elaborates on the methods used to construct the machine learning solution to this problem and indicates different candidate models for the problem. The 4th section illustrates our findings for the constructed models' performance on this problem and the corresponding dataset. The 5th section is the conclusion section of the project which summarizes our findings and offers further ways of improvements. The 6th section is the resources that we've consulted for some machine learning knowledge.

2. Problem Formulation

This problem can be modeled as a machine learning problem. A data-point in this model is a day. The prediction of the model is based on some relevant features which specify a day. They are the previous 7 days' bitcoin value (closing price in US dollars). Label (quantity of interest) of the model is tomorrow's bitcoin value (closing price in US dollars).

Feature	Description	Selection Reason	Measurement
Day-1	Bitcoin value 1 day ago (closing price $)	Next bitcoin value is correlated with the bitcoin value 1 day ago.	Past data is known and can be retrieved from online databases.
Day-2	Bitcoin value 2 days ago (closing price $)	Next bitcoin value is correlated with the bitcoin value 2 days ago.	Past data is known and can be retrieved from online databases.

Day-3	Bitcoin value 3 days ago (closing price $)	Next bitcoin value is correlated with the bitcoin value 3 days ago.	Past data is known and can be retrieved from online databases.
Day-4	Bitcoin value 4 days ago (closing price $)	Next bitcoin value is correlated with the bitcoin value 4 days ago.	Past data is known and can be retrieved from online databases.
Day-5	Bitcoin value 5 days ago (closing price $)	Next bitcoin value is correlated with the bitcoin value 5 days ago.	Past data is known and can be retrieved from online databases.
Day-6	Bitcoin value 6 days ago (closing price $)	Next bitcoin value is correlated with the bitcoin value 6 days ago.	Past data is known and can be retrieved from online databases.
Day-7	Bitcoin value 7 days ago (closing price $)	Next bitcoin value is correlated with the bitcoin value 7 days ago.	Past data is known and can be retrieved from online databases.

3. Method

Data can be easily retrieved from online databases for bitcoin which keep track of the bitcoin value every day. The database that is used for this project is retrieved from https://www.coindesk.com/price/bitcoin which provide date (year-month-date), closing price ($), 24h open ($), 24h high ($), and 24h low ($) from 10/01/2013 to 11/03/2021. That corresponds to 2708 data points for the model which is quite sufficient to train and test the model, because it complies well with the rule of thumb: 10*|features|= 70 << 2708 [MLBook].

Since the problem concerns numerical values, regression models are considered for this project. The hypothesis space is chosen as linear regression and polynomial regression (linear reg. hypo. space is a subset of polynomial reg. hypo. space with deg=1). The approach was to start with simple models and then try more complex models in the hypothesis space. Therefore, it was plausible to start with the linear model, then polynomial of degree=2, then polynomial of degree=3 and then their results were compared. After polynomial of degree=3, there was no point of going further since it dramatically overfitted and turned out to be too complex for this problem. The loss function that is used was mean squared error (MSE) loss which is the number one choice for many regression models [MLBook] and it was easy to implement with scikit-learn library [ScikitLearn].

This project is implemented by using python and scikit learn ML libraries [ScikitLearn]. Since data is sequential and 7 sequential features are needed, one datapoint is preprocessed into batches of length 8, namely 7 separate previous days' features and 1 next day's label. Then, data is split into 3 disjoint sets: training, validation, and test sets with the ratio of 0.6, 0.2, 0.2 respectively. It was a trivial task with scikit-learn [ScikitLearn]. Firstly, the data set is single split into training and valid & test sets with the ratio of 0.6, 0.4 respectively. Then combined valid & test sets are split into separate valid and test sets with the ratio of 0.5, 0.5 respectively. Training

set is commonly used to train these 3 hypotheses. Validation set is used to evaluate the performance and choose the best hypothesis among the 3 hypotheses. Finally, the test set is used to evaluate the final performance of the chosen hypothesis. Then 3 separate hypothesis functions are created with scikit-learn: linear reg (deg=1), polynomial reg deg=2, and polynomial reg deg=3 [ScikitLearn]. Then models are trained and their results are compared accordingly.

4. Results

Hypothesis	Training Error (MSE)	Validation Error (MSE)	Training R2 Score	Validation R2 Score
Linear Reg.	171936.6	161639.4	0.996517573314	0.99582997994596
Polynomial Reg degree=2	1615693.5	1495366.1	0.967275545375	0.96142212306122
Polynomial Reg degree=3	18060920.0	1004077800.0	0.634191897092	-24.9034845863439

The MSE of linear regression on both training and validation sets is significantly (almost 10 and 100 times respectively) lower than the corresponding MSE of the polynomial regression models. Linear regression is the absolute winner in this experiment, since the success and the performance metric of a ML model is based on the smallest validation error.

However, in this case one can question the performance of the linear regression either, because in all of these models including linear regression MSE losses are very high. This phenomenon is because of the fact that there are over 2700 data points and variation is significant, because data is collected from 2013 till 2021 (bitcoin price was roughly 100 $ in 2013 and 48000 $ in 2021). This phenomenon also explains why training error is slightly more than validation error (in the case of linear reg and polynomial reg of deg=2), because the training set is much larger than the validation set (exactly 3 times larger) and bearing in mind the high variation this is possible.

In this regard, R2 score is used to observe the correlation of the model and the data. The R2 score of the linear regression is again the absolute winner compared to polynomial regression, because it's roughly 0.99 and significantly closer to 1 in both training and validation sets.

Additionally, polynomial regression of deg=2 would be the second best model choice, because its R2 score is roughly 0.96 in both training and validation sets which is not that bad. However, it's not the case for polynomial reg of deg=3, because validation error (MSE) is 100 times worse than the training error which simply indicates that it overfits. Furthermore, its R2 score implies

the same interpretation which is that it does better in the training set than the validation set, namely overfitting problem.

Consequently, linear regression is the chosen model, because of its dramatic performance in terms of MSE and R2 score compared to polynomial regression. Hence, the performance of the chosen linear regression is finally evaluated on a disjoint test set which is independent from training and validation sets and also its final performance result complies well with the previous results closely.

Model	Test Error (MSE)	Test R2 Score
Linear Regression	212259.33	0.9962554148885925

Moreover, this model can be further improved by normalization which would ease gradient descent to converge faster and more accurately. Also, filtering out outliers and stabilizing variation in the dataset could potentially decrease the loss in the model.

5. Conclusion

Linear regression and polynomial regression models (deg=2, deg=3) are compared with respect to their validation error and R2 score. The best model is chosen with the lowest validation error and highest R2 score which significantly turned out to be linear regression. Then, linear regression's final performance is evaluated on a separate test set which has the same data size as the validation set. Its test error (MSE) and R2 score on the test set is consistent and more or less the same compared to the results for the training and validation sets.

This model in this form can be further improved with normalization and outlier filtration. Although pure 7 dimensional linear regression produces promising results, this problem can be modeled more accurately with special models designed specifically for time varying processes in economics i.e. autoregressive models: autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA) [AutoReg].

6. References

[MLBook] A. Jung, "Machine Learning. The Basics", 2021, mlbook.cs.aalto.fi

[ScikitLearn] Introduction to machine learning with python: A guide for data scientists. O'Reilly Media.

[AutoReg] Introductory Time Series with R (2009th ed.). New York, NY: Springer.