Bitcoin Closing Price Prediction

Lakshya Kumawat (B20AI019)

1. Introduction:

Virtual currencies are a form of cryptocurrency which is an impressive technical achievement in digital marketing, nevertheless. Virtual currencies live on, and they couldn't fully replace fiat or conventional currencies. In the current study, we are trying to show an interesting new perspective from the view of economics questions surrounding currency governance, the characteristics of money, political economy of financial intermediaries, and the nature of currency computation.

Primarily, the main challenge of the bitcoin exchange rate is its high rate of price fluctuation. The objective of our current study is to forecast the bitcoin price with improved efficiency using machine learning models and minimizing the risks for investors as well as policy-makers.

2. Dataset:

The Dataset contains the opening price, Closing price, Lowest price, Highest price, Volume and Market Cap. Our main focus is to predict the closing price of the next day. The Dataset has all these prices of 1556 Days from Apr 28, 2013 to July 31, 2017. We have to divide the dataset into train and test. So, we take training data 70% of the total dataset and the remaining data is for testing, without shuffling, because we want the data day-wise continuous.

3. Methodology:

Overview

There are various classification algorithms present out of which I shall implement the following:

- Decision Tree Regressor
- XGBRegressor
- Linear Regressor
- Random Forest
- LSTM

Exploring the Dataset and pre-processing:

There is no null value present in the dataset except in the Volume column, but we dropped it. So, there is no problem.

- Drop all the columns except the Closing price column.
- Convert the date into numerical format.
- Sort the data according to date.
- Generate five features corresponding to each closing price, which are the previous five days closing price.
- Scale the data using MaxMinScaler, whose formula is $X_{new}=(X - X_{min})/(X_{max} - X_{min})$.
- Split the data into test and train data linearly, not random.

Implementation of various regression algorithms:

- **Decision Tree Regressor**: The decision tree classifier creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute. I tune various hyperparameters like: *criterion = 'squared_error', max_depth, max_leaf_nodes, min_sample_split, random_state=42.*

- **XGBoost**: XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. Boosting is an ensemble modeling technique which attempts to build a strong classifier from the number of weak classifiers. It is done by building a model using weak models in series. First, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models are added. Two models implemented:
  - XGBoost Deep (max_depth=12)
  - XGBoost Shallow (max_depth=2)

- **Linear Regressor:** Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. It is a linear approach for modeling the relationship between a scalar response and one or more explanatory variables.

- **Random Forest:** Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the

random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Hyperparameters taken are: *max_depth=6, random_state=0,n_estimators=500*

- **LSTM:** stands for Long-Short Term Memory. Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn. All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tan h layer.

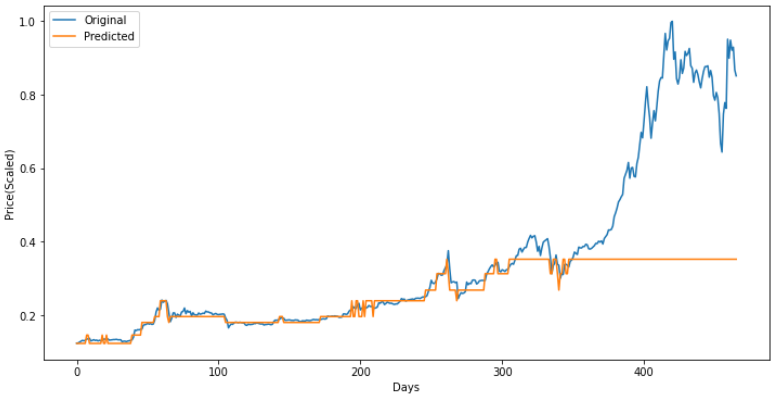
Evaluation of Models:

We can evaluate our models on different parameters. Here I am going to evaluate using the R2 score parameter.

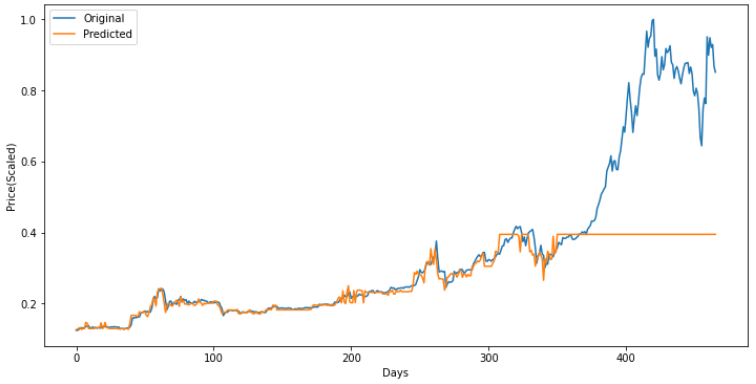
Model	R2 Score
Decision Tree	0.2793
XBG(shallow)	0.4038
XBG(deep)	0.3721
Linear Regression	0.9926
Random Forest	0.2206
LSTM	0.9918

Plots:

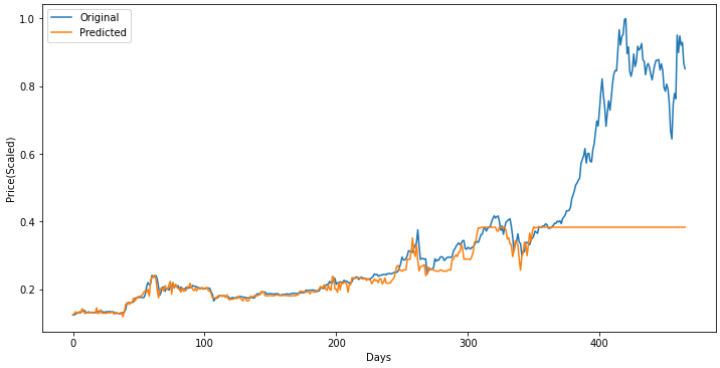
Decision Tree



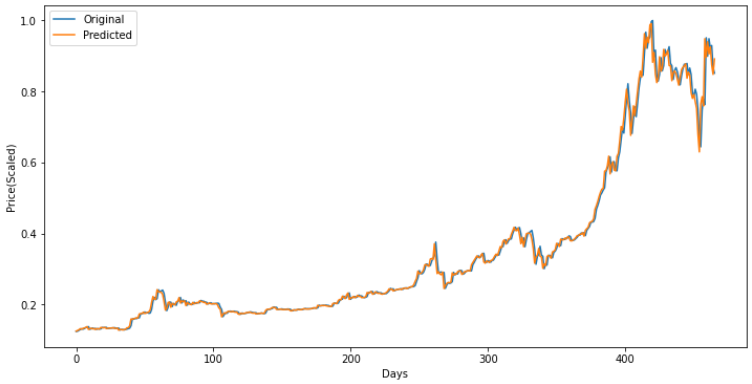
XGB(shallow)



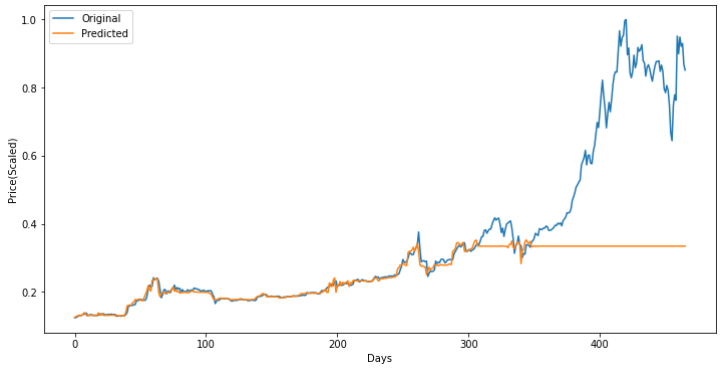
XGB(deep)



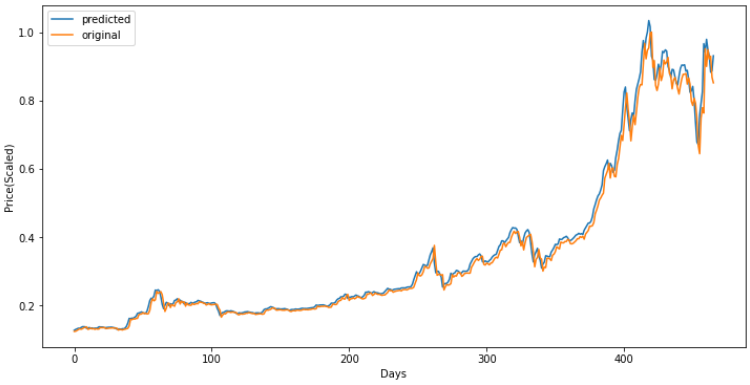
Linear Regression



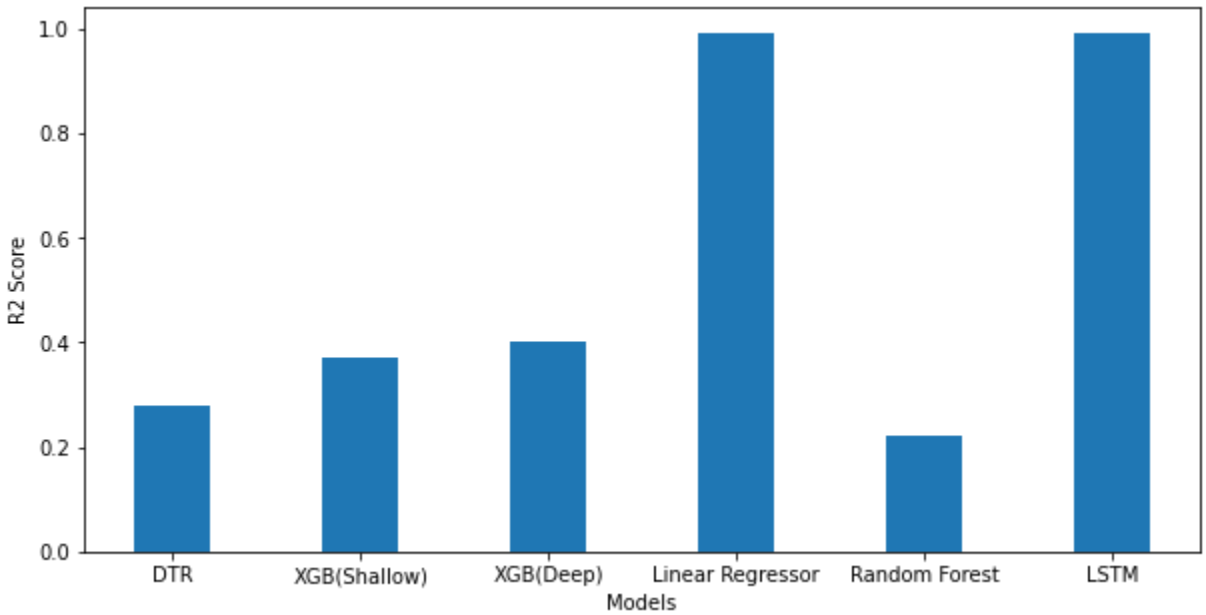
Random Forest



LSTM



Model vs Accuracy Plot:



Results & Analysis:

- Maximum r2_score we get on Linear Regression.
- We get approximately the same r2_score for Linear Regression and LSTM.
- All models other than these two are working very badly.
- LSTM is working well because it can remember inputs over a long period of time. It contains information in a memory like a memory in a computer.
- Linear Regression is working very well, and does not overfit, even its chance of overfitting is very high.
- When there is drastic change in price, other models are not able to predict them properly and converge to a constant value.
- I am taking the past 5 days values as features for the next day. If I take more days like 30, the models are not working well or working the same. It means the older values have less importance than the newer ones.

We conclude that Linear Regression and LSTM are the best models for bitcoin prediction.
But…
We were successful to a good extent in predicting Bitcoin prices with some of our models. But, in general, this will not work well in the real world. This is because history might not repeat itself. Also, new factors affecting Bitcoin's price can come into play with time !!!

References:

- Sklearn Documentation - for models, algorithms and their attributes and parameters.
- Cryptocurrency price prediction using LSTMs (towardsdatascience.com)
- Keras -Tutorial The Ultimate Beginner's Guide to Deep Learning in Python