Cryptocurrency Price Prediction

Ali Ahmad, Niro Srishan, Sardar Haider Waseem Ilyas, Renato Tizon

Github: https://github.com/renatito1701/TimeSeriesFinalProject

The Context

Definition:

Cryptocurrency is decentralized digital money that's based on blockchain technology. It is a medium of financial exchange that is digital, encrypted, and decentralized.

Problem Statement:

Prices of cryptocurrencies remain extremely volatile and are difficult to predict for investors. This is an attempt to understand the underlying factors involved in determining price. The end-goal is to use our knowledge of time series models to create sophisticated and accurate forecasts.

Assumptions

Data:

Unlike trading stocks and commodities, the cryptocurrency market is not traded on a regulated exchange. The market remains open 24/7 and is accessible through a growing number of exchanges.

We will be modeling our data around the closing prices for any particular day which over a substantial period of time should not deviate from the daily patterns, trends and averages.

Modeling

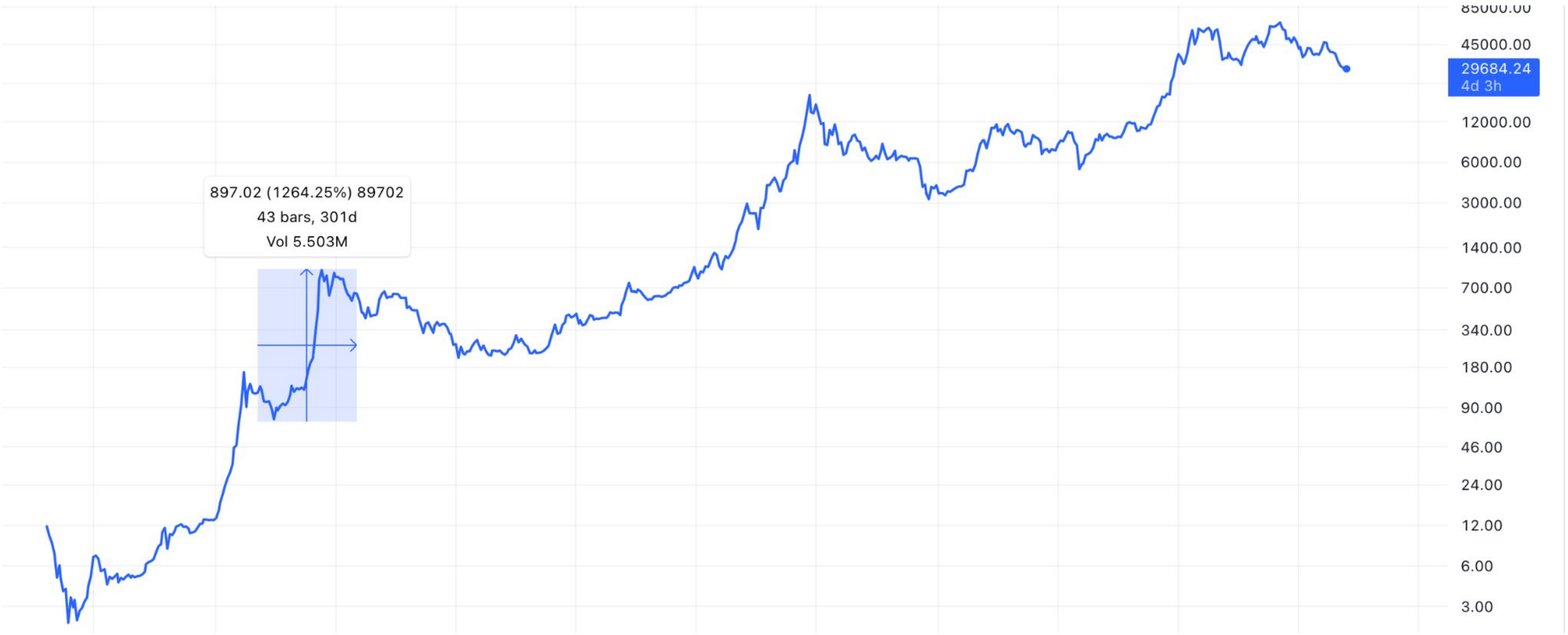
The feasibility of models will be examined via an analysis on bitcoin prices - one of the most prevalent cryptocurrencies. The model architecture should be generalizable to cryptocurrencies with significant trading volumes.

When building our models we treat the data as time series models, under the assumption that this framework will yield better results. [1]

Our models will have reduced accuracies as the time-frame for forecasts increase.

Exploratory Data Analysis

Bitcoin Market Fluctuations



Bitcoin Price Variation

Tests for Stationarity

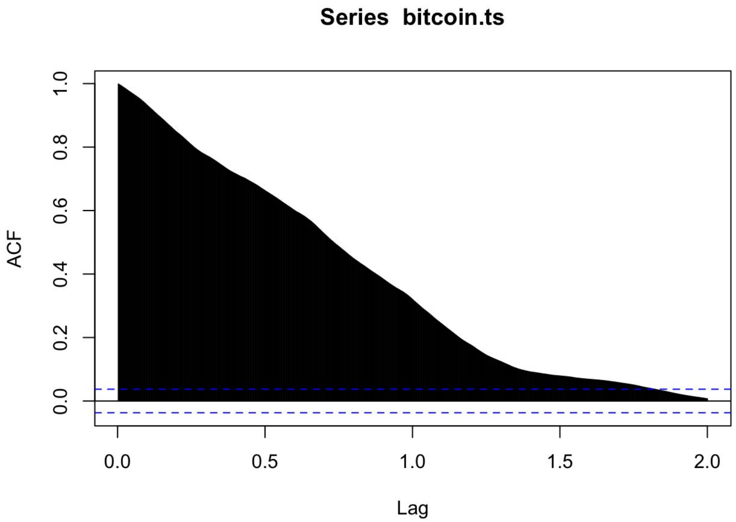
```
> adf.test(btc$Close)

        Augmented Dickey-Fuller Test

data:  btc$Close
Dickey-Fuller = -2.2656, Lag order = 14, p-value = 0.4659
alternative hypothesis: stationary

> kpss.test(btc$Close)

        KPSS Test for Level Stationarity

data:  btc$Close
KPSS Level = 17.718, Truncation lag parameter = 9, p-value = 0.01
```

Closing Price is not stationary

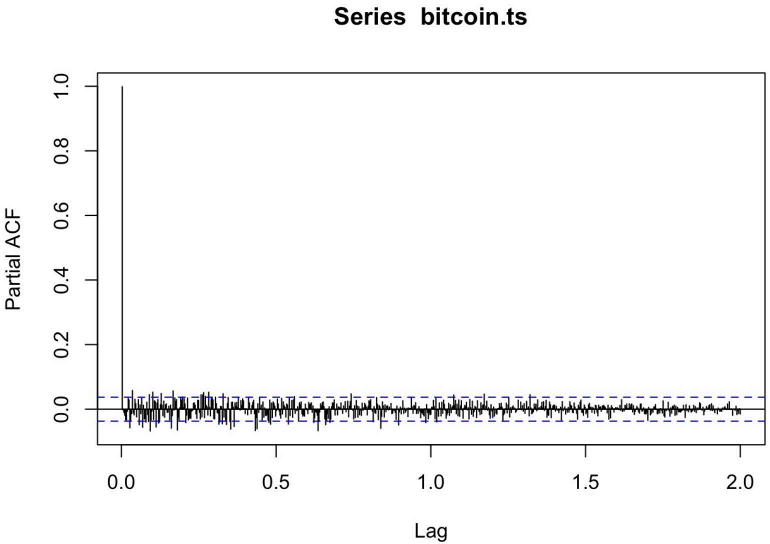
```
> adf.test(diff)

        Augmented Dickey-Fuller Test

data:  diff
Dickey-Fuller = -13.456, Lag order = 14, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(diff) : p-value smaller than printed p-value
> kpss.test(diff)

        KPSS Test for Level Stationarity

data:  diff
KPSS Level = 0.076557, Truncation lag parameter = 9, p-value = 0.1
```

First Difference of Closing Price is stationary

Bitcoin - ACF and PACF

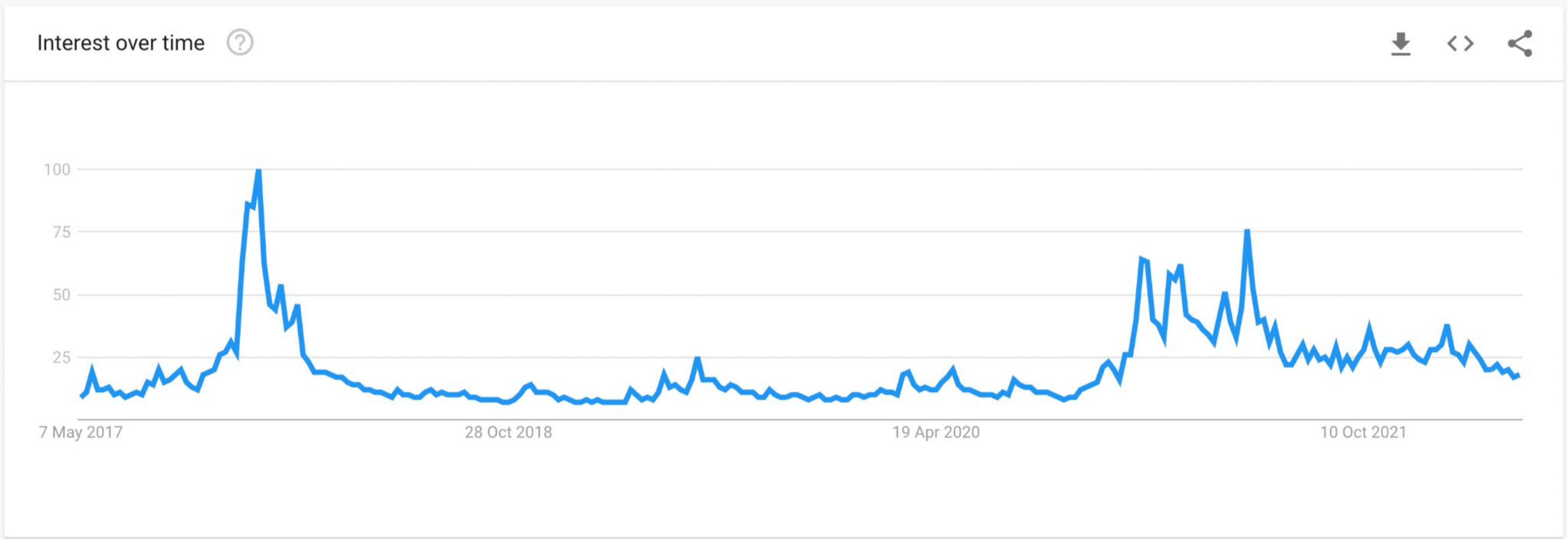


ACF Tests for Closing Price



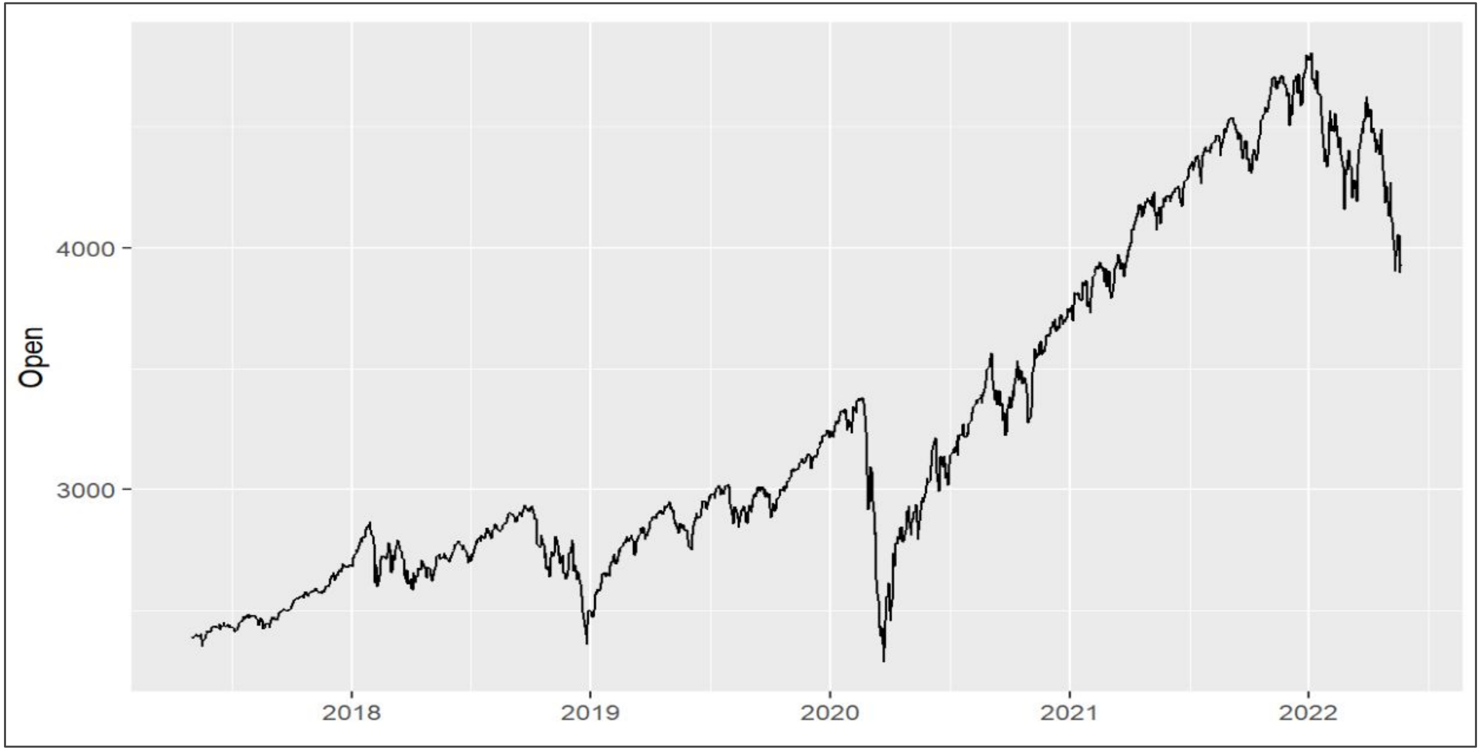
PACF Test for Closing Price

Bitcoin - Google Trends

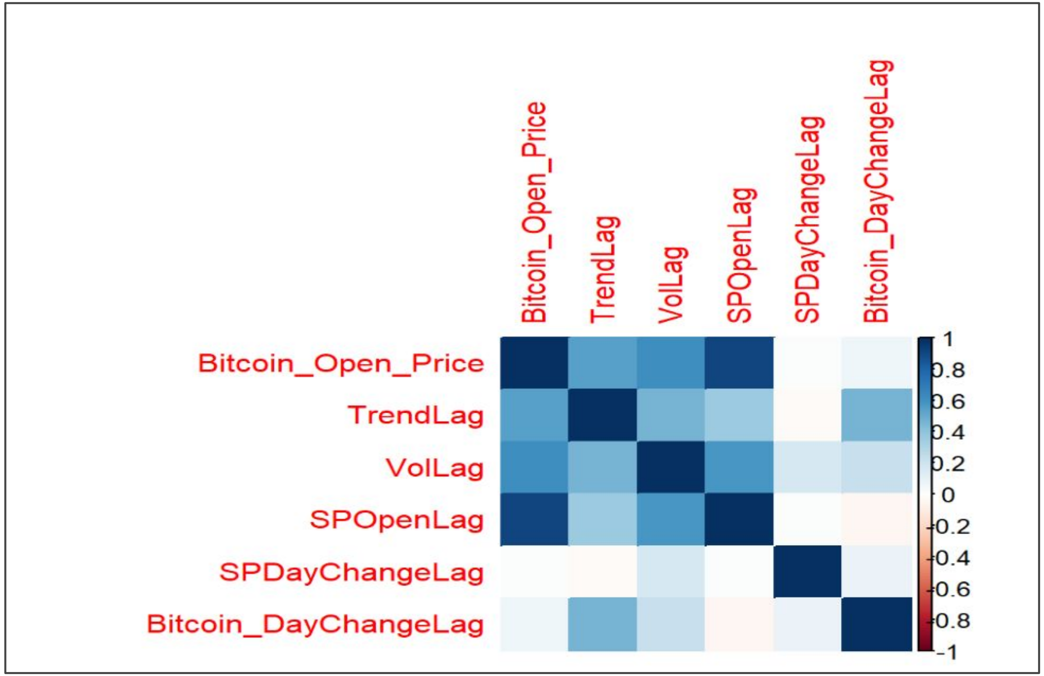


Google Trends Data for Bitcoin: May 2017 - April 2022 [4]

S&P 500



Correlation Plot



Data Processing and Feature Engineering

Deep Dive into Data

Data Processing and Feature Engineering:

1. Creation of XTS objects for closing price and trading volume to fit time series models
2. Compute lag values for Google trends, trading volume, price volatility
3. Split into train and tests to create forecasts for a 30 day interval
4. Log(Price) - utilize the logarithmic scale to account for massive and sudden changes in value
5. Create variable to account for direction of previous day price change

All Exogenous Variables:

1. Trading Volume
2. Volatility in Daily Trading Volume
3. Google Trends Data for Bitcoin
4. S&P 500 Open Index
5. Volatility in S&P 500 Index (Max - Min)
6. Lag Closing Price
7. Lag Google Trends
8. Lag Price Volatility

All Endogenous Variables:

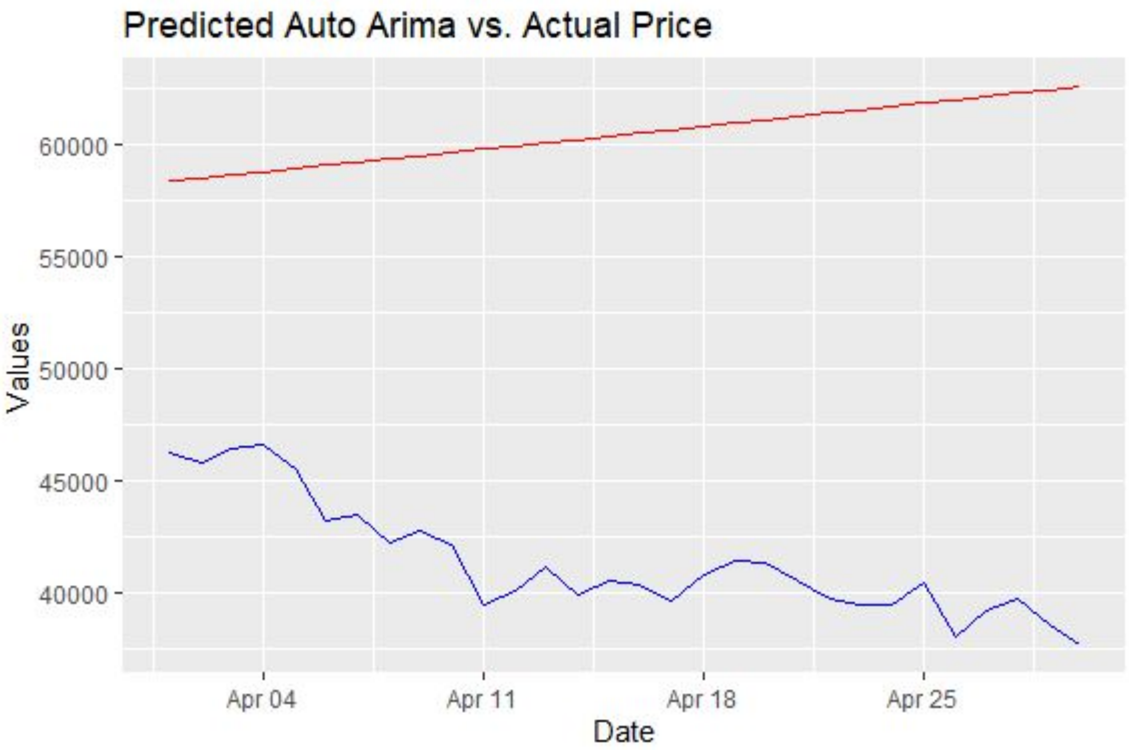
1. Bitcoin Closing Price
2. Log Closing Bitcoin Price

Model Engineering

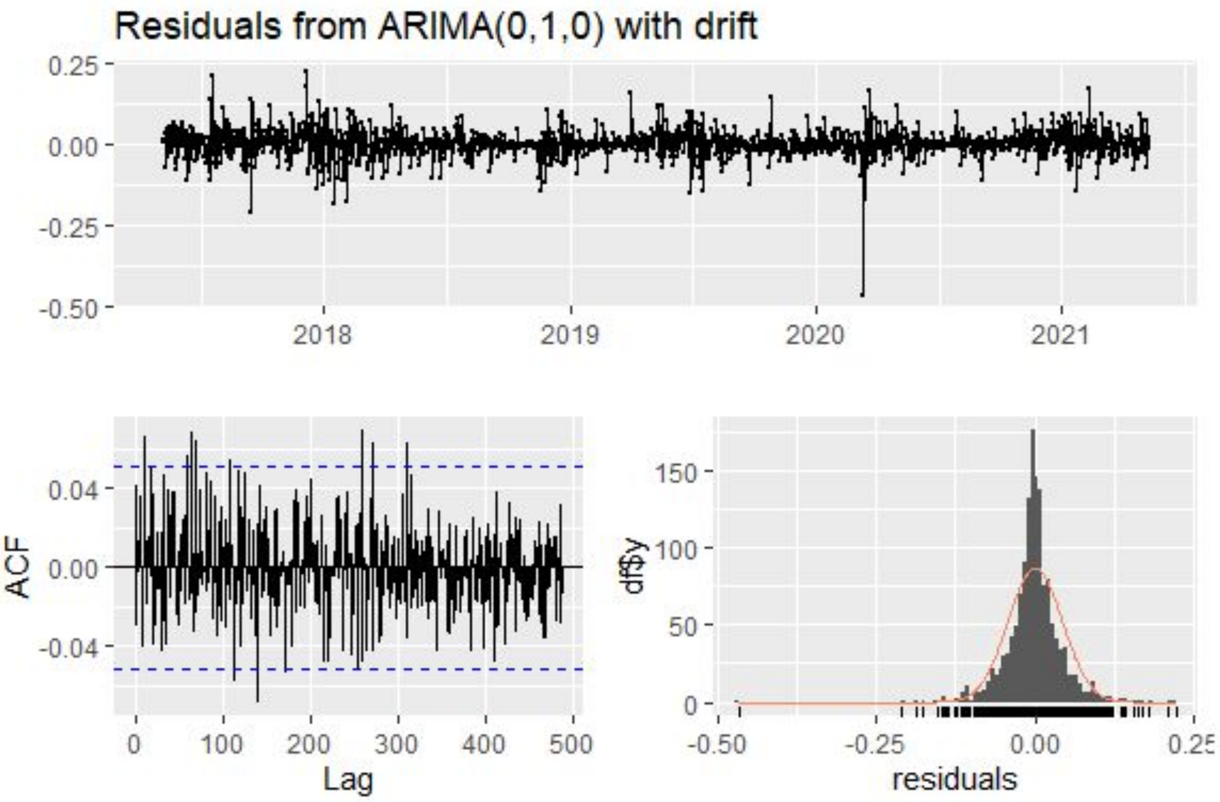
Comparison between Models using Log Price

Model.Name	AIC	Smape	R Squared	RMSE
Arima (0, 1, 1)	-5053.019	0.0317	NA	0.347
Arima (0, 1, 2)	-5054.001	0.0317	0.124	0.348
Arima (1, 1, 2)	-5054.126	0.0322	0.793	0.365
Auto Arima	-5056.649	0.0351	0.768	0.387
Seasonal Arima	-5056.649	0.0351	0.768	0.387
Auto Arima - Xreg(Vol Change)	-5052.820	0.0317	0.004	0.347

Auto ARIMA Forecast (Log Price)



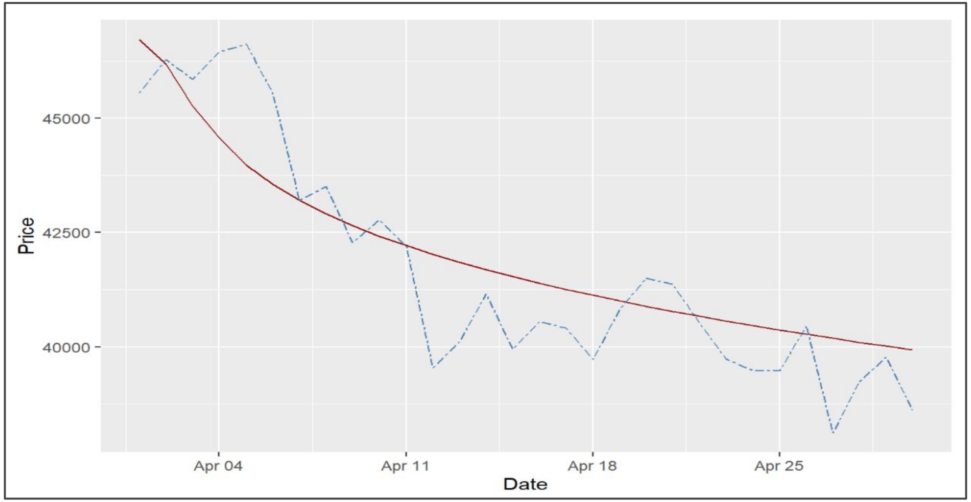
Residual Analysis - Auto ARIMA (Log Price)



Comparison between Models using Regular Price

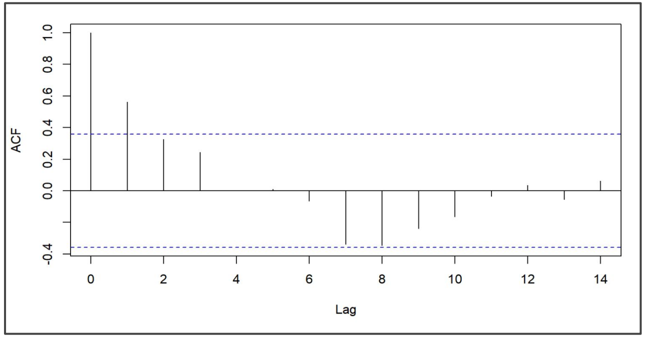
	AIC	Smape	R Squared	RMSE
ARFIMA (0,.5,5)	30119.3	0.02	0.82	1187.43
ARIMA - S&P (0,.5,5)	29717.5	0.11	0.84	5467.83
Prophet	–	0.09	0.69	4143.97
LSTM		0.084		2271.89

Bitcoin Forecast - ARFIMA (Normal Price)

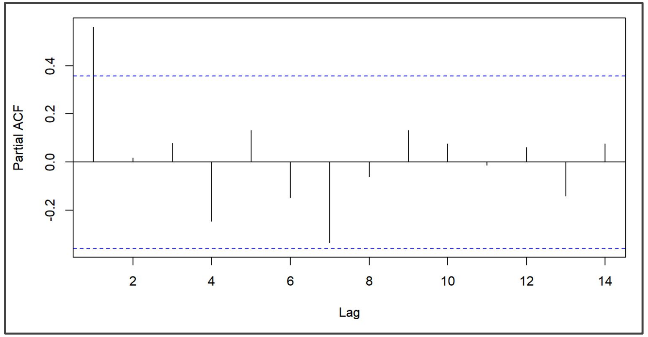


ARFIMA

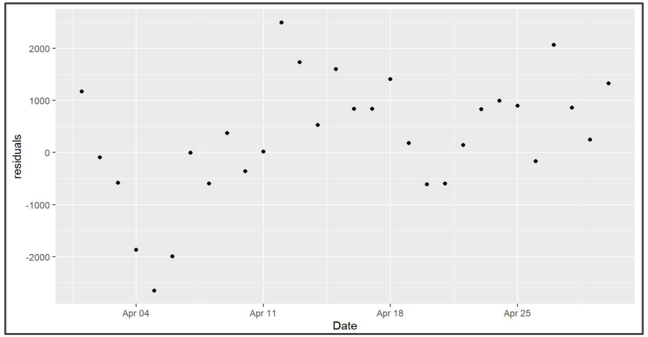
Residual Analysis - ARFIMA



ACF

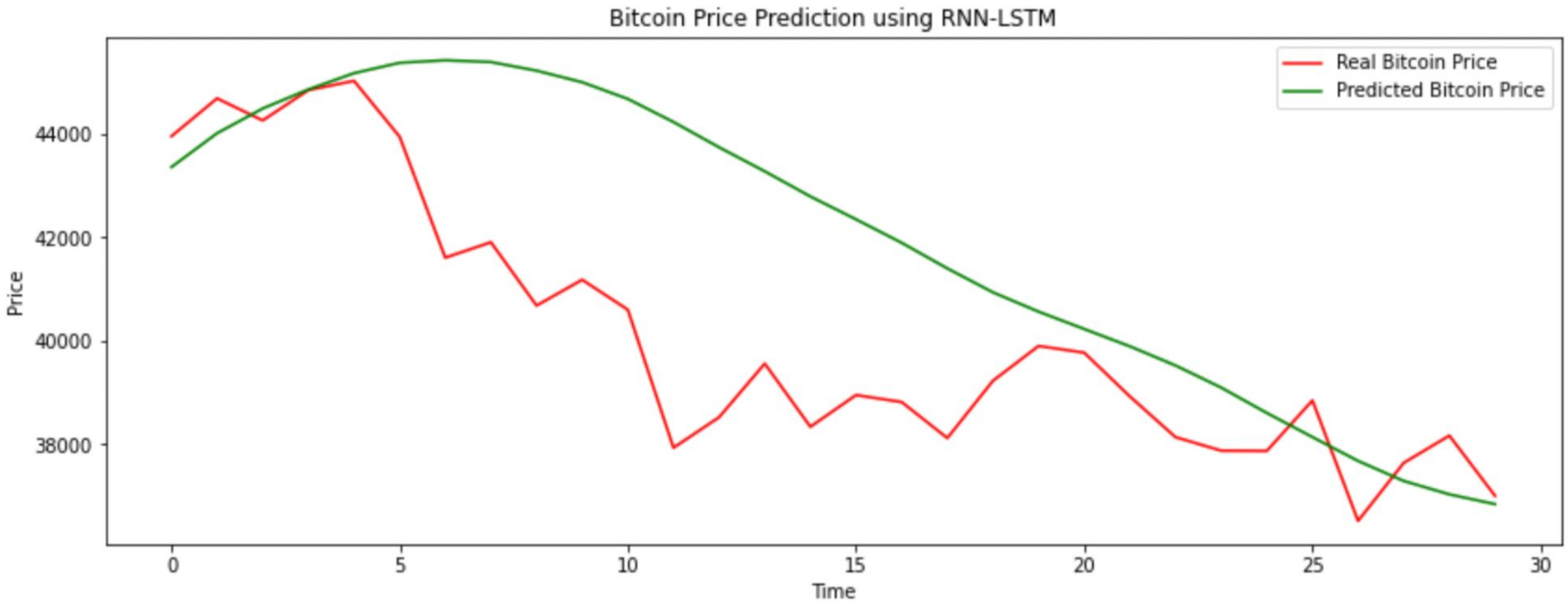


PACF



Residuals

Bitcoin Forecast - LSTM (Normal Price)



Model Evaluation and Limitations

Log Price vs. Linear Price

- When analyzing the stock market technical analysts typically view price scale in both linear and logarithmic scale.
- Logarithmic price scales are usually better than linear price when severe price increases or decreases occur, drastic changes in price are perfect for a logarithmic scale.
- Bitcoin is ideal given its high volatility and extreme price changes in the last year, ranging from a low **29796.29** on July, 2021 to a high of **68000+** in November 2021.

Model Limitations

- When we try to train ARIMA models to a 6/7-years-long period, in which bitcoin price has experienced extreme increases and decreases in price, we observe that it introduces large prediction errors.
- The sharp fluctuation in price calls for more features to be extracted and used along with the closing price for a more accurate prediction of the price
- ARIMA is quite efficient in making predictions in short spans of time; but as the time grows, the precision rate would decrease.
- Our models were not able to handle external events such as the disruption of markets vis a vis the COVID pandemic.

Why predict bitcoin prices using ARIMA?

- The ARIMA model has been widely utilized in banking and economics since it is recognized to be reliable, efficient, and capable of predicting short-term share market movements.
- ARIMA models are mainly used for demand forecasting but can also be used to predict the future price of stocks based on the past prices. Given this application, it can seamlessly be applied to bitcoin given its non-stationary nature and despite its general higher volatility.
- Recent applications of ARIMA for bitcoin prediction shows that while it doesn't not perform great on long time periods, it outperforms recurrent neural networks in short term time periods.

Future Work

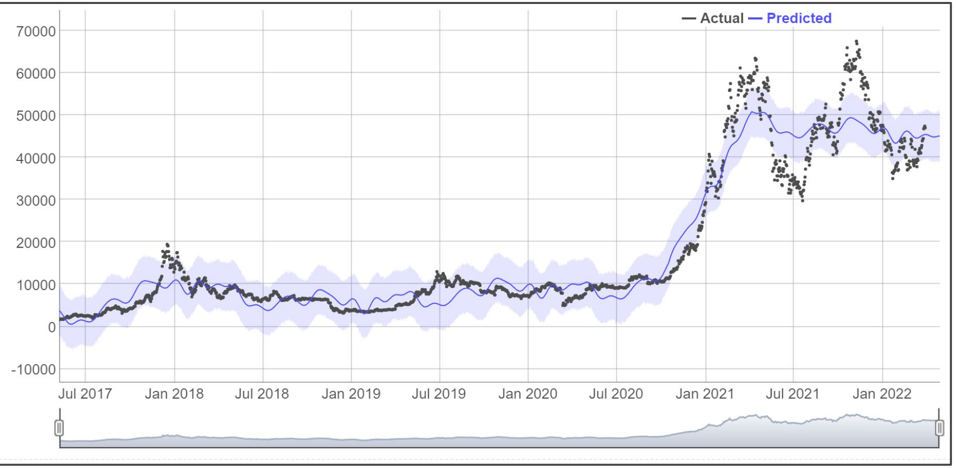
- Use short-term time periods to train model given ARIMA’s weakness for long term prediction
- Incorporate data (Open, Close, Volume) for some of the most popular cryptocurrencies such as Ethereum, Solana, Ripple’s XRP to help predict Bitcoin prices.
- Incorporate data from other sources such as Bitcoin supply on exchanges
- Add sentiment analysis in forecasting of prices
- Utilize multi-variate models to forecast correlated variables
- Have model run in real time and make predictions

Appendix

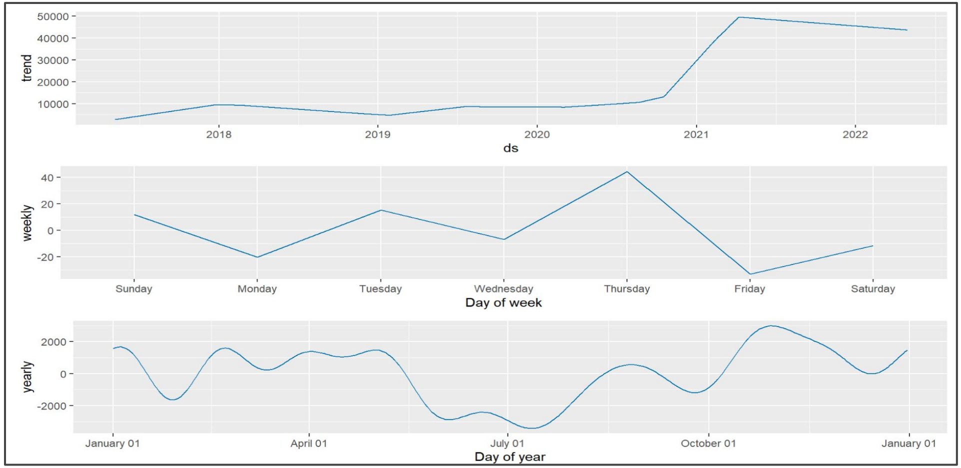
References

1. https://s3.ap-northeast-2.amazonaws.com/journal-home/journal/jips/fullText/200/jips_680.pdf
2. https://www.coindesk.com/price/bitcoin/ - Used for EDA and visualization of bitcoin prices
3. https://coinmarketcap.com/currencies/bitcoin/
4. https://trends.google.com/trends/explore?date=2017-05-01%202022-04-30&q=bitcoin
5. https://www.diva-portal.org/smash/get/diva2:1261399/FULLTEXT01.pdf
6. Manoharan, Poongodi & Vijayakumar, Vignesh & Chilamkurti, Naveen. (2020). Bitcoin price prediction using ARIMA model. International Journal of Internet Technology and Secured Transactions. 10. 396. 10.1504/IJITST.2020.10028982, https://www.researchgate.net/publication/340961010_Bitcoin_price_prediction_using_ARIMA_model
7. Bitcoin price prediction using ARIMA and LSTM, Yiqing Hua, E3S Web Conf. 218 01050 (2020), DOI: 10.1051/e3sconf/202021801050, https://www.e3s-conferences.org/articles/e3sconf/pdf/2020/78/e3sconf_iseese2020_01050.pdf
8. Bitcoin Price Prediction: An ARIMA Approach, Amin Azari, KTH Royal Institute of Technology, https://www.diva-portal.org/smash/get/diva2:1261399/FULLTEXT01.pdf

Bitcoin Forecast - Prophet



Prophet



Prophet Seasonality

LSTM Layers and Parameters

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 60, 50)            11600

 dropout (Dropout)           (None, 60, 50)            0

 lstm_1 (LSTM)               (None, 60, 60)            26640

 dropout_1 (Dropout)         (None, 60, 60)            0

 lstm_2 (LSTM)               (None, 60, 80)            45120

 dropout_2 (Dropout)         (None, 60, 80)            0

 lstm_3 (LSTM)               (None, 120)               96480

 dropout_3 (Dropout)         (None, 120)               0

 dense (Dense)               (None, 1)                 121

=================================================================
Total params: 179,961
Trainable params: 179,961
Non-trainable params: 0
_________________________________________________________________
```