Dogecoin Cryptocurrency's Analysis and Forecast

Completed for the Certificate in Scientific Computation
Spring 2023

Ahmad Jasim Azizi
Bachelor's of Science and Arts in Mathematics
The University of Texas at Austin

Dr. William Bulko
Assistant Professor of Instruction
Department of Computer Science

Abstract

I collected data from Yahoo! Finance for the cryptocurrency called Dogecoin and ran some analysis on it in order to see if I can forecast the price of it. I utilized various software and tools to get the results which I needed. Some of these tools are Google Colab, Anaconda Software, Jupyter Notebook, R-Studio, and the microsoft office suite such as MS Excel and MS Word. I also used programming languages such as Python and R in order to further my analysis with the data that I collected from Yahoo! Finance.

The Prophet method that I used tests the general direction of Dogecoin based on the data being trained for one month, one year, and maximum duration of the data available on Yahoo! market.   This does not, yet forecast the price of the Dogecoin within minutes; however, it does show the general direction of the stock, which is more crucial as of what is floating on the news and on the social media of famous influencers such as Elon Musk.

The reason that I used Prophet is that cryptocurrencies are simply too unpredictable unless there is an extended period of time with detailed historical observations provided. Dogecoin has had multiple strong seasonalities which has happened for the past decade, which is beyond the time-frame of Elon Musk's tweets. These events are previously known important, yet irregular, events.

Introduction and Conclusion

Artificial buys and sells are common when it comes to the cryptocurrency market. When there is a sudden change in the normal distribution of the data, this is essentially considered an artificial buy or sell (i.e. pump and dump). When there are anomalies in the numbers that are provided by a cryptocurrency, it can be utilized to make enormous amounts of profits by those who are savvy with the stock market.

Initially, Dogecoin started as a joke given it mainly replicated Bitcoin over a decade ago. However, this “meme coin” eventually hit the news charts after Elon Musk tweeted about it on Twitter shortly after the Covid-19 pandemic. As of today, this so-called “meme coin” has over $12 billion in market capital and over 800 million Dogecoins in volume.[1]

Although Dogecoin’s volume is fairly consistent unless there is some type of news on it in the mass media, the drastic change in the volume is often correlated to Mark Cuban, Elon Musk, and other famous entrepreneurs and game-changers who are constantly looking to change the way we do things in the world. Most of its volume has been concentrated in the lower portion of the scatterplot, which means that despite the massive pumps, the dump is inevitable, at least as of current trends. While these anomalies should not be underestimated for further analysis as they are directly correlated with the pumps and dumps that happen in the market, for the purpose of this project, they are used as a leverage for other anomalous and eventually non-anomalous activities in the cryptocurrency market. In fact, these anomalies set the pace of the cryptocurrency.

The general trend of Dogecoin is dependent on many factors but upon doing a deep analysis of the data based on different seasons which I divided into monthly, yearly, and maximum time periods, I was able to get to a conclusion that the overall forecast of the cryptocurrency is positive; however, it is crucial to mention that there are many factors that could lead into this forecast not holding true such as collapse of the general cryptocurrency market, influences by influential entrepreneurs, and any interference by powerful parties such as governments.

[1] “Dogecoin Price: Doge Price, USD Converter, Charts.” https://crypto.com/price/dogecoin. Accessed April 20, 2023. https://crypto.com/price/dogecoin.

Materials and Methods

For us to understand Facebook's Prophet model for forecasting, we need to understand how time series works in general and what type of trends are used with it. Time series processes can be decomposed into three parts:

1) the trend, 2) the stationary component, and 3) noise.

The trend component accounts for changes in the nature of the time series over time. Time series processes with trends are non-stationary. The mean, variance, or both are a function of time. We need to properly account for trends in dynamic processes in order to test hypotheses with time series data.

Deterministic VS Stochastic Trends:

In general, a trend is considered deterministic where realizations of the time series process are a fixed function of time, such as a high-order polynomial

$$y(t) = \beta 0 + \beta 1t + \beta_2 t^2 + \beta_3 t^3$$

Clearly, in this case, $E(y_t)$ depends on t. Now, if we add a stationary component to the trend, for example,

$$y(t) = \beta 0 + \beta 1t + \beta_2 t^2 + \beta_3 t^3 + A(L)\varepsilon_t$$

The process is said to be trend-stationary. Long-run forecasts will converge to the trend. In the simplest case, we have yt = β0 + β1t, a linear trend, which can be expressed as

$$\Delta y_t = \beta_0, \ or\ \Delta y_t = \beta_0 + \varepsilon_t\ (with\ noise)$$

If the trend is deterministic, to isolate the stationary component, we detrend the data by regressing $\{y_t\}$ on a high-order polynomial function of time. The order of the polynomial can be determined by *t*-tests and *F*-tests as well AIC and SBC measures of fit.[2]

Spurious Regressions

It should be evident that two time series that have deterministic trends will correlate spuriously. Now, if the true data generating process is as follows:

$$Y_t = Y_{t-1} + \varepsilon_{yt}$$
$$Z_t = Z_{t-1} + \varepsilon_{zt}$$

[2] https://sites.pitt.edu/~jch61/PS2740/Notes_and_Slides/TSCS_Week5_Trends.pdf

and we estimate the regression

$$Y_t = \beta z_t + e_{yt}$$

Thereafter, e_{yt} will have $f_y(t)$, that will correlate with z_t through $f_z(t)$. The estimate of β will suffer from the left out variable bias. Furthermore, give the $\{e_{yt}\}$ are dependent, the standard error estimates will eventually be biased as well.

Facebook's Prophet Model:

I used a model called Prophet which was developed by Facebook. It is a decomposable time series model (Harvey & Peters 1990)[3] that has three main model parts which are 1) trend, 2) seasonality, and 3) holidays. They are joined in the below equation. Importantly, it is also designed to have easy to understand parameters that can be fixed without understanding the details of the underlying model. Facebook's Prophet model could be considered a regression model which is nonlinear and it has the following form

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t$$

Here, *g(t)* describes a piecewise-linear trend (or "growth term"), *s(t)* describes the different seasonal patterns, *h(t)* stores the holiday effects, and finally ε_t is the white noise error term[4]. Moreover, *g(t)* is the trend function which models non-periodic changes in the value of the time series, *s(t)* represents periodic changes (e.g., weekly and yearly seasonality), and *h(t)* represents the effects of holidays which occur on potentially irregular schedules over one or more days. The error term t represents any idiosyncratic shifts that are not accommodated by the model itself. I will make the parametric assumption that t is normally distributed.

This definition is quite similar to a generalized additive model (GAM) (Hastie & Tibshirani 1987)[5], a class of regression models with possible non-linear smoothers applied to the regressors. Here we use only time as a regressor but possibly several linear and nonlinear functions of time as components. Modeling seasonality as an additive component is the same approach taken by exponential smoothing (Gardner 1985)[6]. Multiplicative seasonality, where the seasonal effect is a factor that multiplies $g(t)$, can be accomplished through a log transform.

- The seasonal component consists of Fourier terms of the relevant periods. In general and by default, order 10 is utilized for yearly seasonality and order 3 is used for weekly seasonality.

---

[3] Harvey, A. & Peters, S. (1990), 'Estimation procedures for structural time series models', Journal of Forecasting 9, 89–108.
[4] Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician*, *72*(1), 37–45.
[5] Hastie, T. & Tibshirani, R. (1987), 'Generalized additive models: some applications', Journal of the American Statistical Association 82(398), 371–386.
[6] Gardner, E. S. (1985), 'Exponential smoothing: the state of the art', Journal of Forecasting 4, 1–28.

- Also, holiday effects are additive as plain dummy variables.
- Overall, the model is estimated with a Bayesian approach. This gives room for automatic selection of the changepoints and other model characteristics.

It is crucial to note that automatic forecasting has a long history and various methods tailored to specific types of time series (Tashman & Leach 1991, De Gooijer & Hyndman 2006[7]). The changepoints (aka, knots) for the piecewise-linear trend are automatically chosen, that is if it is not explicitly specified. As another option, logistic function can be used to set an upper bound on the trend.

While at first glance the Prophet model seems simple and not mathematical enough, it is merely a mirage. Hence, we need to dive deeper into it by asking the following question:

What do the functions g(t), s(t), and h(t) exactly mean? And what is ε_t?

Simply speaking, the model represents the following functions:

$$\begin{aligned} y(t) \ = \ & piecewiseTrend(t) + \\ & seasonality(t) \ + \\ & holidayEffects(t) \ + \\ & i.i.d.\ noise \end{aligned}$$

However, if we were to dive deeper into each function, we notice that there is, in fact, a lot of mathematics involved in the Prophet model as a whole. For instance, the ***g(t)* trend** models *non-periodic* changes (i.e. growth over time). This refers to the trends or changes over a long period of time. The procedure provides two possible trend models for g(t), "a saturating growth model, and a piecewise linear model."

$$g(t) \ = \frac{C}{1+e^{-k(t-m)}}$$

with *C* is the carrying capacity, *k* is the growth rate, and *m* is an offset parameter. Suppose there are *S* changepoints at times sj, j=1,...,S, Prophet defines a vector of rate adjustments;

$$\delta \ \in \ \Re^{S}, \ where\ \delta\ is\ the\ change\ in\ rate\ that\ occurs\ at\ times\ sj$$

***s(t)* seasonality** presents *periodic* changes (i.e. weekly, monthly, yearly). This refers to seasonality or periodic or short term changes. The seasonal component *s(t)* provides an adaptability to the model by allowing periodic changes based on sub-daily, weekly, yearly seasonality.

[7] De Gooijer, J. G. & Hyndman, R. J. (2006), '25 years of time series forecasting', International Journal of Forecasting 22(3), 443–473.

Prophet relies on Fourier series to provide a malleable model of periodic effects. *P* is the regular period the time series will have. Approximately arbitrary smooth seasonal effects are therefore tied in with a standard Fourier series; I rely on the Fourier series to provide a flexible model of periodic effects (Harvey & Shephard 1993)[8]. Let P be the regular period we expect the time series to have (e.g. $P = 365.25$ for yearly data or $P = 7$ for weekly data, when we scale our time variable in days). We can approximate arbitrary smooth seasonal effects with

$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right)$$

Fitting seasonality requires estimating the 2N parameters

$$\beta = [a1, b1, ..., aN,\ bN]^T$$

This is done by constructing a matrix of seasonality vectors for each value of t is our historical and future data, for example with yearly seasonality and $N = 10$:

$$X(t) = \left[\cos\left(\frac{2\pi(1)t}{365.25}\right), \ldots, \sin\left(\frac{2\pi(10)t}{365.25}\right)\right]$$

Meaning the seasonal component is;

$$s(t) = X(t)\beta$$

In the generative model, Prophet takes $\beta \sim Normal(0,\ \sigma^2)$ to impose a smoothing prior on the seasonality.

Truncating the series at *N* applies a low-pass filter to the seasonality, so, albeit with increased risk of overfitting, increasing *N* allows for fitting seasonal patterns that change more quickly. For yearly and weekly seasonality we have found *N=10* and *N=3* respectively to work well for most problems. The choice of these parameters could be automated using a model selection procedure such as AIC.

Moreover, ***h(t)*** ties in effects of **holidays** (on potentially irregular schedules ≥ 1 day(s)). This refers to the effects of holidays to the forecast. The component h(t) speaks for predictable events of the year including those on irregular schedules; however, for the sake of this project, *h(t)* is not used because cryptocurrencies are tradeable 24/7..

[8] Harvey, A. C. & Shephard, N. (1993), Structural time series models, in G. Maddala, C. Rao & H. Vinod, eds, 'Handbook of Statistics', Vol. 11, Elsevier, chapter 10, pp. 261–302.

And finally, ***e(t)*** covers idiosyncratic changes not accommodated by the model. This refers to the unconditional changes that are specific to a business or a person or circumstance. It is also called the error term or simply the "noise". This leads us to ***y(t)***, which is simply the **forecast** given the previous parameters and functions are satisfied.

Libraries, Programming Languages, and Tools:

The main languages used in my research project were Python and R. I also used Google Colab to make use of different Python libraries in order to analyze and visualize my data. I utilized Pandas open source data analysis and manipulation tool, which is also built on top of the Python programming language.

Furthermore, I had to take advantage of libraries such as colorama, pystan, ephem, prophet, matplotlib, pyplot, networkx, seaborn, numpy, datetime, and so on to get the desired results for my project. And as mentioned earlier, for my data, I mainly used the historical Dogecoin datawhich is provided by Yahoo! Finance.

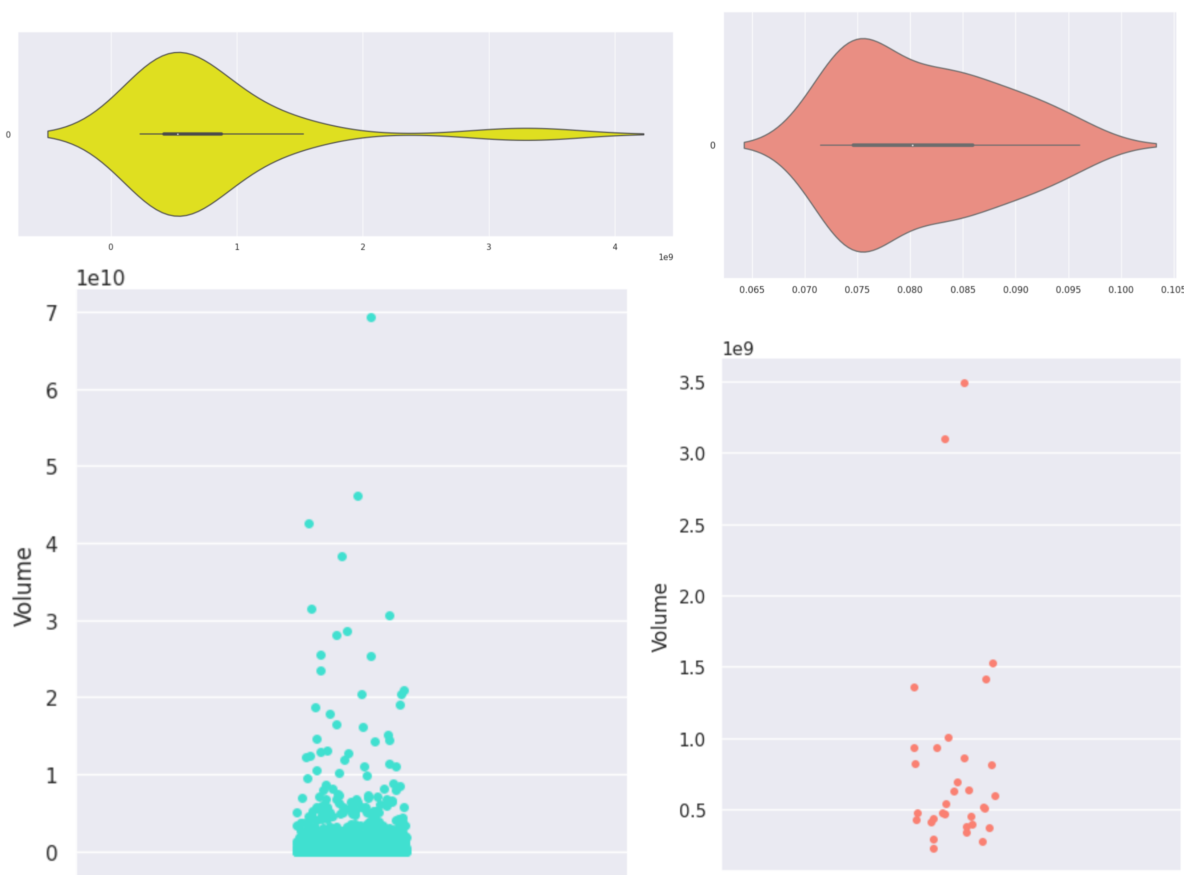
Finally, I made use of a few books that helped me understand the math and statistics behind the Prophet model. Some of these books that helped me the most were *Practical Data Science with R by Nina Zumel and John Mount, an Introduction to Statistical Learning with Applications in R by G. James and Daniela Witten, and Predictive Analytics by Eric Siegel*. It was crucial for me to further my knowledge of these topics given this is my first ever research project.

Results

Given the nature of the model that I used, I relied heavily on different time-frames of the data which I pulled from Yahoo! stocks and had to make many inferences in order to make sense of it. For example, when it comes to Dogecoin cryptocurrency, it is vital to visualize the necessary data in different time-frames because of the sudden pumps and dumps that happen in its volume. These abrupt changes often set the pace for the direction of the cryptocurrency for the next few days and at times weeks to come.

I decided to split my work into monthly, yearly, and maximum time periods of the data that I used with my data-frames. The monthly data suggests that upon an anomalous detection of the volume in the price, there are further artificial buys which happen within hours after the first buy. In the following violin and scatter plots, we can see that most of the concentration is around the first bigger pump. Hence, the bigger the pump, the more volume around this buy.

Yearly and Monthly plots of opening prices and volume:

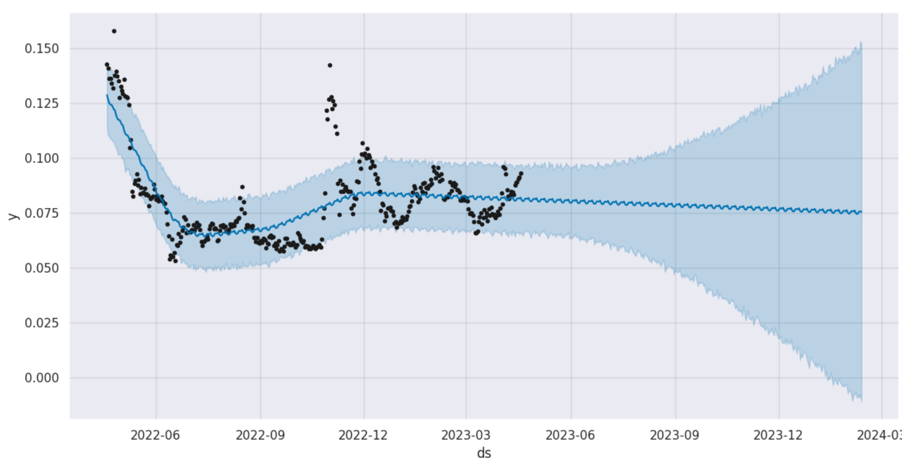


We can further see that the market for Dogecoin behaves similarly to that of Bitcoin, in that there are massive buyouts and sellouts at different time periods but in general a lot of the density is concentrated between 0.01 to .1 cents.

Maximum plots and forecast:

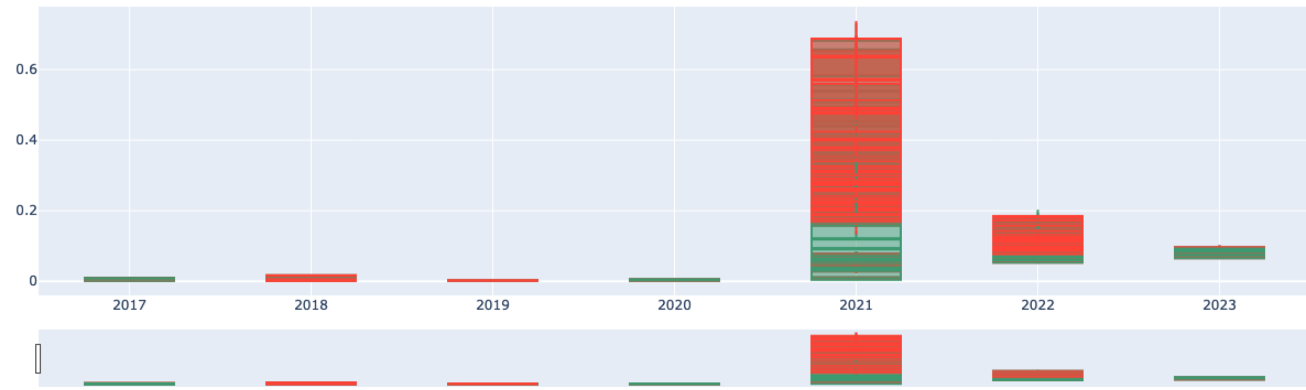
Using the Prophet model, I was able to detect the noise and separate the different time periods of the data to get to a conclusion that the overall trend in which Dogecoin cryptocurrency is going is positive. But does that mean that we are absolute about our prediction?

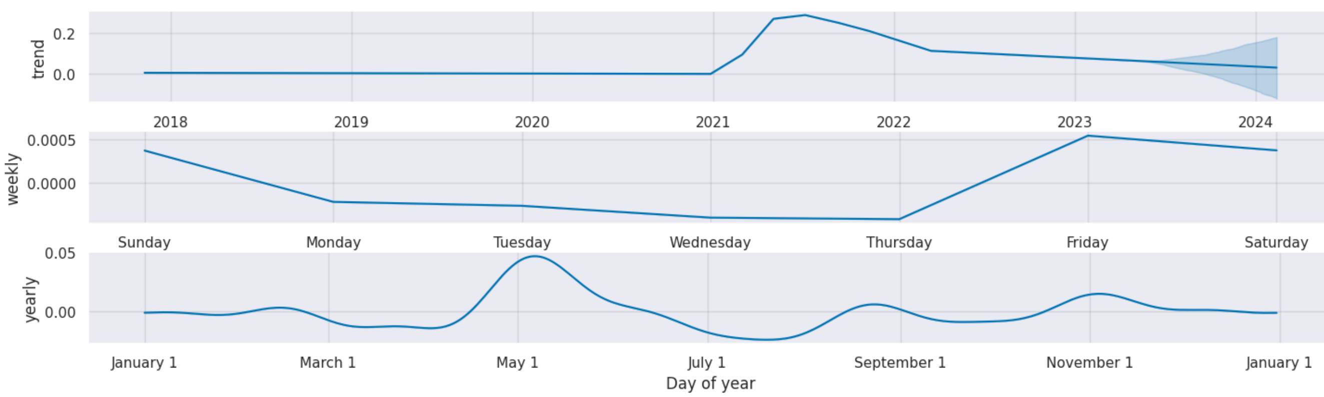
The answer is no! In fact, we need to understand that using this model, we are forecasting the general direction of this cryptocurrency more than predicting it. However, despite the upticks and downticks that happen to it on a regular basis, the slope is definitely upwards. It is, however, important to point out that the forecast in this research paper is mainly focused on the long-run trend of this cryptocurrency.



Furthermore, I found out that there is a lot more activity on Dogecoin which begins on Thursday evenings and the slope goes down by Sunday evenings. yearly and monthly prices of

Dogecoin. In general, the shorter the time-frame of the data, the wider the spectrum for the prediction of the slope for Dogecoin. We can see from the following plots that the direction of the trend of the cryptocurrency is depending on the volume, opening price, and the seasonality (in this cash time of the year and day of the week).



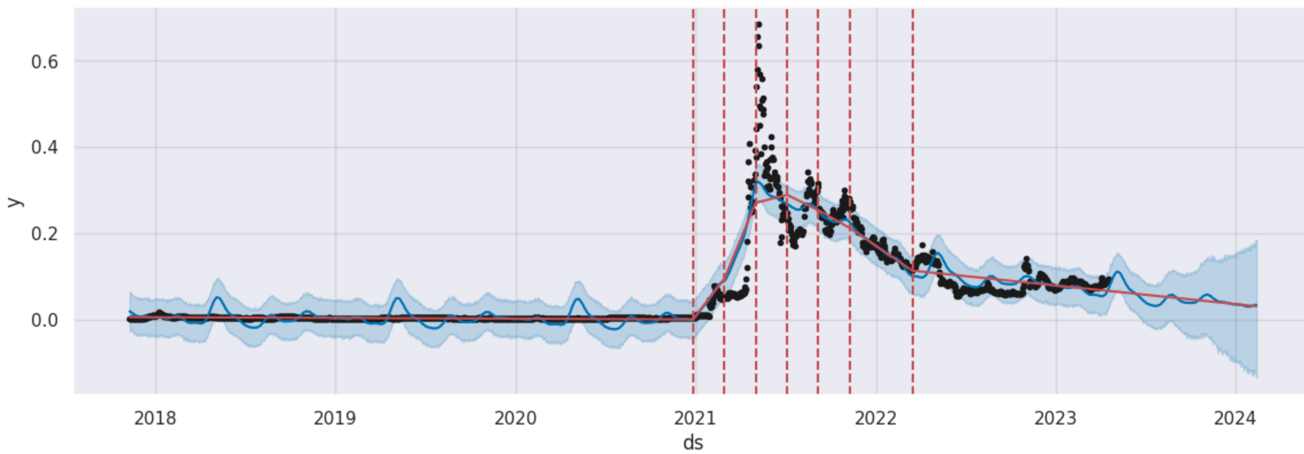


Furthermore, when it comes to changepoints, I detected a few crucial knots for the past 4-5 years. The following are the major changepoints that occurred in the price of Dogecoin from January 2018 through March 2023 in table form:

Dogecoin Crypto's Changepoints

64	2018-01-12
191	2018-05-19
254	2018-07-21
318	2018-09-23
445	2019-01-28
572	2019-06-04
635	2019-08-06
762	2019-12-11
826	2020-02-13
953	2020-06-19

1080	2020-10-24
1143	2020-12-26
1207	2021-02-28
1334	2021-07-05
1461	2021-11-09
1588	2022-03-16



Mean Absolute Percent Error (MAPE)

The mean absolute percentage error (MAPE), also known as mean absolute percentage deviation (MAPD), is a measure of prediction accuracy of forecasting method in statistics and it often expresses the accuracy as a ratio defined by the following formula:

$$MAPE = \frac{\%100}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|$$

Where A_t is the actual value and F_t is the forecast value. Their difference is divided by the actual value A_t. The absolute value of this ratio is summed for every forecasted point in time and divided by the number of fitted points n. Given its intuitive interpretation in terms of relative error, that means absolute percentage error is commonly used as a loss function for regression problems. Given its limitations for forecasting for practical applications, the model makes use of other measures proposed in literature: mean absolute error (MAE), mean squared error (MSE) symmetric mean absolute percentage error (SMAPE), root mean squared error (RMSE), etc, which we need not dive deeper for the purpose of this research project.

The following table represents the six different types of metrics given each time horizon by taking the moving average (i.e. over 37 days in this case).

	horizon	mse	rmse	mae	mape	mdape	smape	coverage
0	37 days	9.714130e-08	0.000312	0.000185	0.058559	0.026369	0.056472	0.673516
1	38 days	9.845843e-08	0.000314	0.000187	0.059759	0.027050	0.057778	0.668950
2	39 days	9.900961e-08	0.000315	0.000189	0.060478	0.026369	0.058658	0.671233
3	40 days	9.973430e-08	0.000316	0.000190	0.061212	0.026369	0.059538	0.666667
4	41 days	1.003544e-07	0.000317	0.000191	0.061752	0.025786	0.060270	0.664384
...	...	...	...	...	...	...	...	...
324	361 days	0.000120	0.010938	0.004191	0.314491	0.046153	0.367125	0.328767
325	362 days	0.000116	0.010753	0.004136	0.314187	0.046153	0.368031	0.328767
326	363 days	0.000108	0.010401	0.004008	0.314005	0.046304	0.369198	0.324201
327	364 days	0.000104	0.010217	0.003871	0.313416	0.046304	0.369308	0.324201
328	365 days	0.000105	0.010227	0.003892	0.313068	0.048526	0.369424	0.324201

And as we can see that MAPE measures the size of the error in percentage terms in the following plot. It is calculated as the average of the unsigned percentage error.

Discussion

One of the reasons that Facebook's Prophet model works is because despite being simple, it is a modular regression model that often works well with default parameters, which allows analysts to select the components which are relevant to their forecasting problems and easily make adjustments as needed.

Facebook's Prophet is essentially "framing the forecasting problem as a curve-fitting exercise" rather than looking explicitly at the time based dependence of each observation. A few points to make about the Prophet model is that it does not allow non-Gaussian noise distribution. In Prophet, noise distribution is always Gaussian and pre-transformation of y values is the only way to handle the values following skewed distribution. The Prophet does not take autocorrelation in residuals into account.

Since the epsilon noise portion in the formula assumes i.i.d. Normal distribution, the residual, is not assumed to have autocorrelation, unlike the ARIMA model. Prophet does not assume stochastic trend; in fact, Prophet's trend component is always deterministic + possible changepoints and it won't assume a stochastic trend unlike ARIMA. Overall, I would like to further my research into understanding the Prophet model more. I would also like to run these analyses on other models to see what results I would get.

Acknowledgements

Special thanks to:

Professor Angela Beasley
Assistant Professor of Instruction
Department of Computer Science

Professor Eric J. Staron
Assistant Professor of Instruction
Department of Mathematics

Professor Sally Ragsdale
Assistant Professor of Instruction
Statistics and Data Sciences

Computer Code

https://github.com/jasimazizi

https://github.com/jasimazizi/dogecoin-crypto-analysis

Reflection

While conducting this research project, I have learned about quite a few changepoint detection methods. In order to finalize what method I want to end up with, I had to study many other models that I was unaware of such as support vector machines, bayesian changepoint finder, changefinder, etc. However, upon understanding the abrupt and sporadic changes with cryptocurrency, hence, Dogecoin, I decided that a more deterministic model is necessary.

This is because short-term pumps and dumps might help us predict Dogecoin but this is also relevant to the reasoning behind the pump or dump. Therefore, a deterministic model such as Prophet might have a better overall result given the parameters used in this research paper. Scraping data by major influencers and online social media and communication platforms is necessary.

Overall, I learned quite a lot about different libraries in Python that are crucial for data analytics and visualization. For example, I increased my proficiency with Pandas, NumPy, and Matplotlib. I also made use of NetworkX and other graph libraries in order to make inferences between different parts of my data.