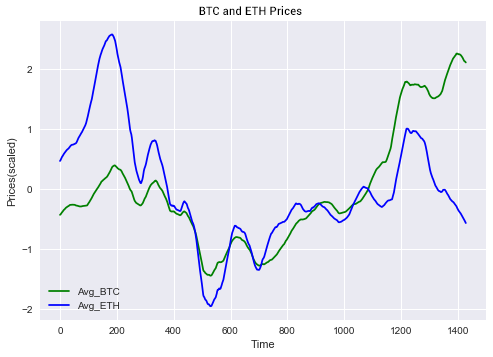
Ever since Bitcoin’s price skyrocketed, there has been constant hype surrounding the crytocurrency market. Alternate coins keep popping up everyday- some are scams, some make it to the top coin list in months. The topic comes up everywhere, whether it’s on the radio, Twitter, Facebook, or at the Thanksgiving dinner table with your grandfather. The people involved are usually fueled by speculation, hoping to come into windfall from the surging market. Just when you think the hype can’t get any more severe, this project is an intersection between crytocurrencies and the glorious maCHEEN learning- the magical word all tech CEOs use to hype up their products. So, I hope this gets you just as hyped as I am. This project uses historical price, cointergration, and macroeconomic covariates to predict a coin future price. The out-of-sample prediction was acceptable for the first ~100 hours. Let’s dig into it.

When talking about classical time series analysis, we believe that an observed time series is a combination of pattern and some random variations. Using this approach, future values are predicted based on its historical data. This method works well in most cases, but what if we’re looking at a time series that is more random than pattern-like? What if a time series is purely speculative and heavily based on current events rather than some rhythmic components? You know…. something we all have heard of- crypocurrency prices.

So if not just some simple pattern, what drives cryptocurrencies prices? Speculations? Innovation? Legal issues? Public opinion? Supply/demand? Bitcoin popularity? Some rich (wo)man decided to buy a million coins last night? Okay, enough speculating speculations (haha). I’ll stop talking and show you the data now.

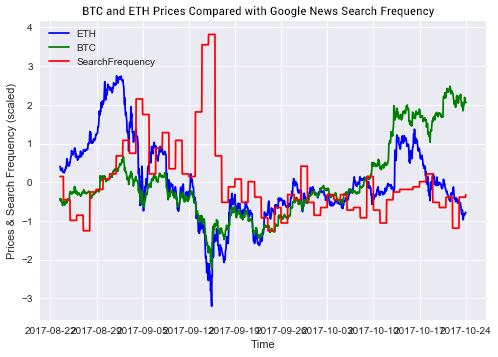
**So what affects crypto prices?**

1. Most people say Bitcoin is the answer. Blockchain technology is a decentralized database system that’s first implemented by Bitcoin. Created by a mysterious person (or group), Blockchain has a very high tendency to transform modern day business operation models. As Bitcoin gains more traction, people keep coming up with alternate coins that are also based on Blockchain technology. So pretty much Bitcoin is the mother of all cryptocurrencies because they came up with the cool technology first. That’s why I think it makes sense that when Bitcoin spikes, every other coin spikes. When it drops, every other coin drops. Following plot is scaled rolling averages of Bitcoin (green) and Ethereum (Blue).



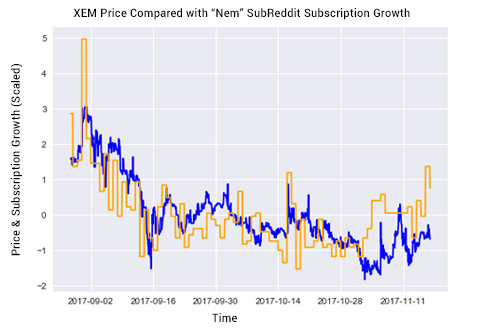
As you can see, the claim looks pretty accurate. But we’re not going to rush into conclusion without statistical methods. Later on, we’re going to talk more about identifying a “cointegrated pair” using Granger Causality Test.

1. It’s also pretty obvious that current events such as legal issues or technological game changers also play a role. Remember when China banned crypto back in September? The price dropped rapidly and everything was in chaos. In an attempt to capture important events like this one, Google News search frequency data is obtained from Pytrends API. Plot below shows spike in search frequency of the word “cryptocurrency” (red) as cryto prices drop.



Interesting right? The search terms used in this project are selected using the Google Keyword Tool. Not only does the tool let you know how popular a search term is, it also suggests a list of related keywords. Using the list provided and the Pytrend API, search frequency data of seven different keywords is obtained. I’ll elaborate more about these terms in later section.

1. Another factor that stands out to me is public perception. The more buy-ins, the more demand and thus, the higher the price. Capturing this data is a bit painful. The paid Twitter API is everything I needed, but I’m unemployed so I’d rather save the money for groceries. Instead, the site redditmetics.com plots out historical subscription growth data of just about any subreddit in the world. So web scraping it is!!! The plot below compares Nem subreddit subscription growth (orange) with Nem historical price (blue).



Just as expected, the subscription growth and the price stick together through the highs and the lows. How cute. I wish someone loves me the way price loves subscription growth.

I hope these visuals are interesting to you. This is just to get you started with some domain knowledge and to introduce you to the problem we’re trying to solve. If the intro didn’t scratch your itch, please check out the complete EDA available on my GitHub here. Next I’m going to jump right into the statistical methods used to build a model that predicts a coin’s future price.

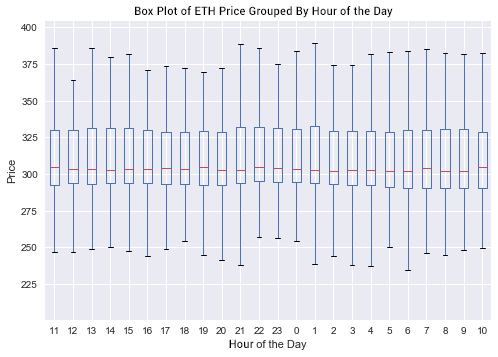
**How did I build the model?**

In this section, we’re going to dig into the methodology. It will be just a summary of each step. If you want to dig deeper into the code, please refer to my GitHub repository here. It’s going to be technical and I’m going to try my best to make it fun and easy to digest. But if you’re not interested in the technical stuff, feel free to skip right into the TLDR section.

1. *Identifying Cointegrated Pair*

A total of 12 top coins’ historical prices over three months period are obtained through Cryptocompare API. But before we can do anything with the time series, we have to make sure that the time series is stationary. To meet the stationary requirements, a time series must have constant mean, constant variance, and constant autocorrelation. Sounds like a lot to ask for, right? In reality, no time series is perfectly stationary. But worry not my friends, Dickey & Fuller have got your back!

* 1. Augmented Dickey-Fuller Test of Stationary. This is a statistical test that allows you to check if the expectations of stationarity are met. It’s a unit root test that determines how strongly a time series is defined by a trend. The test uses an autoregressive model and optimizes an information criterion across different lag values. The null hypothesis is that the time series can be represented by a unit root (means it’s not stationary). Some statistician came up with the magic threshold of 0.05 and a lot of people agreed with the number. So if your p-value is less than 0.05, you could reject the null hypothesis. But then again, the result should be interpreted for a given problem to be meaningful. It turned out that, assuming a threshold of 0.05, the historical prices from all 12 coins don’t pass the stationary test (surprise!). In that case, we’ll have to stationarize the time series and re-test them.
  2. Differencing. This is a popular method used to stationarize time series. It can be used to remove trends and seasonality. Taking the difference of consecutive observations (lag 1) is used for this project. If a time series had a seasonal component, the lag value should be the period of the seasonality. In our case, there is no obvious seasonal component. Box plot below shows how Ethereum hourly average is relatively constant throughout 24 hour of the day. The variance varies, but there is no obvious pattern.



After we performed lag-1 difference on the time series, all 12 time series passed the Dickey-Fuller stationary test! Phew.

* 1. Granger Causality Test. This is a statistical hypothesis test for determining whether one time series is useful in forecasting another. A time series A “Granger-cause” time series B if lag values of A provided statistically significant information about future values of B. In this project, we’re using this test to identify a cointegrated pair- a pair of cryptocurrencies in which one coin’s lag values can be used to predict the other coin’s future values.

Now that the historical prices data of all 12 coins is stationary, we constructed a total of 132 dataframes, each of which is a permutation pair (not to be confused with combination pair) of the 12 coins' historical prices. Yikes, that’s really confusing. So, for example, let’s say I had ETH, BTC, and LTC historical prices data. Then I would need to make 6 data frames: ETH & BTC historical prices, ETH & LTC historical prices, BTC & ETH historical prices, BTC & LTC historical prices, LTC & ETH historical prices, and LTC & ETH historical prices. Notice that ETH & BTC and BTC & ETH aren’t the same thing! This is just to prepare our data for the StatsModels Granger Causality Test. The test’s null hypothesis is that the time series in the 2nd column does not Granger-cause the time series in the 1st column. We need to test whether ETC causes BTH or BTH causes ETC. This is why we need permutation pairs and not combination pairs! After performing 132 tests, DASH & BCH pair was previously selected as the cointegrated pair because of the strongest correlation. However, with further research, it turns out that the strong correlation is due to the surge in Korean trading. Since this is not a normal condition, we instead pick XEM and IOT (Nem and Iota) as our cointegrated pair since it has the strongest correlation under normal condition. For the purpose of this project, IOT historical price will be used as one of XEM future price predictors.

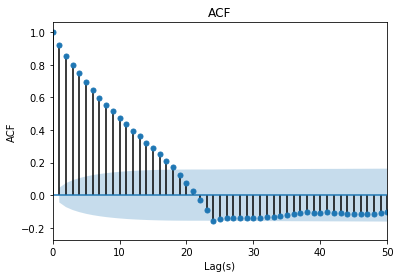
1. *Feature Selection.* 
   1. Querying Data. The following is the acquired data and it’s sources:

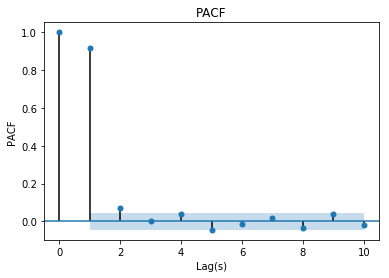
* Cryptocompare API: XEM and IOT historical prices in hour frequency
* Pytrends API: Google News search frequency of the phrase “cryptocurrency”
* Scraping redditmetrics.com: Subreddit “CryptoCurrency”, “Nem”, and “Iota” subscription growth
* Pytrends API: Google search frequency for the phrases “Nem wallet download”, “Iota wallet download”, “Nem price”, Iota price”, Bitcoin price”, “GPU for mining”- these keywords are selected using Google Keyword Tool
* Yahoo Financials API: AMD and Nvidia stock prices- these are the top 2 semiconductor companies used for coin mining
  1. ElasticNet Regularization. As you might have already noticed, the queried data is somewhat related. For example, the Google search frequency of the phrases “Bitcoin price” and “CryptoCurrency” might contain very similar information. Building a model using related features like these creates redundancy and it’s bad! Just like how humans get emotionally unstable (a.k.a irritated) off of redundancy, machines freak out and become unstable due to multicollinearity too! Thank God ElasticNet is a thing. The algorithm linearly combines the L1 and L2 penalties of LASSO and Ridge methods. It “shrinks” the redundant predictors’ coefficients to zero and so it becomes an industry standard way of feature selection. Using ElasticNet result, features that have non-zero coefficients are to be selected. In this project, ElasticNet is performed on 13 predictor variables acquired above, with XEM historical price being the dependent variable. After running the algorithm, we are left with three predictors that have non-zero coefficients. These features will be used to build the final model. Regardless of the ElasticNet result, I’ve also tried building a model using all 13 predictors then compared the result with the model built using the three selected features. It turns out that the difference in performance isn’t significant (MSE = 0.107 for all 15 features and MSE = 0.105 for 3 features). However, I’m going with the three features model still because I’m from LA and they always say “the leaner the better.”

1. *Building the Model.*

For this project, we are going to use ARIMAX model to predict XEM future price. Just like ARIMA model, ARIMAX produces forecasts based on autoregressive (AR) and moving average (MA) terms. However, ARIMAX includes exogenous variables in the model as well. In this case, the three predictor variables previously selected will be used here.

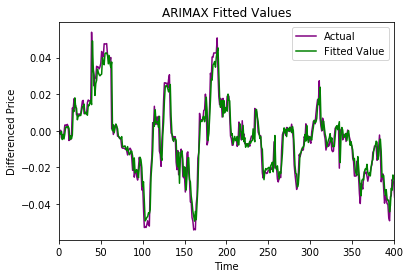
* 1. Data Preprocessing. Since we’ve already talked about stationary and Dickey-Fuller tests in the previous section, I will omit the details here. The data obtained is already standardized prior to ElasticNet, so all we need to do is to perform differencing then make sure it passes Dickey-Fuller test. After that, the data is cleaned and split in to test and train sets.
  2. ACF & PACF. Now that our data is ready, we need to 1) determine if the time series is AR or MA process and 2) determine what order of AR or MA process we need to use in the model. The first question can be answered using ACF. For AR series, the correlation goes down gradually without a cut off value. ACF could also be used to determine the lag order of MA series- it’s the cut off value. However, if we had AR series, PACF cut off value is used to determine the lag order instead. Plots below are the ACF and PACF of XEM historical price.



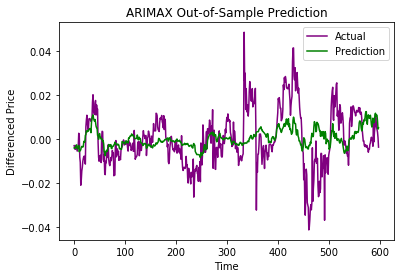


As we can see, it’s an AR process since ACF doesn’t have a cut off value. Then, looking at PACF, it cuts off at lag 1, which would be the parameter we’re using for our ARIMAX model.

* 1. ARIMAX. Using AR 1 and 3 exogenous variables, the plot below is the fitted value compared to actual value.



Using the fitted model, XEM price prediction is obtained. The plot below is the out-of-sample prediction of XEM 600 steps ahead of time.



As expected, the model performs better in the beginning. This is because the prediction errors keep compounding as longer time passed. After around 100 steps, the model doesn’t perform so well. The mean squared error for steps 1-100 is 0.039 while the mean squared error for steps 101-600 is 0.119.

**TLDR**

Iota historical price combined with other macroeconomic covariates such as Google search frequency of the word “Nem price” and “Nem” subreddit subscription growth data was used to build ARIMAX model to predict Nem price. The out-of-sample prediction performance was acceptable for the first ~100 hours. Anything beyond is pretty much junk.

This project is my very first data science project and there is a lot of room for improvements. Getting the paid Twitter data or using different machine learning model might be able to improve the performance significantly. At that point, I might as well come up with a trading signal algorithm and use it to automate trading. But for now, this project will continue to serve as a portfolio piece. And yes, I’m disappointed that I’m not rich off of crypto trading by now. Back to crying now. Adios.

I hope you enjoyed this article just as much as I enjoyed working on it! Leave your comments and let me know what you think.