Predicting Bitcoin Price Action using ARIMA

This study focuses on forecasting bitcoin daily closing prices. It makes use of data collected online with the goal of creating an ARIMA (time series) model with a Mean Absolute Percentage Error of under 20%. This study will use the auto_arima method by the pmdarima Python library for hyper-parameter selection and model build. This study hypothesizes that an ARIMA model can be made with the available data and a MAPE of < 0.20 using daily time steps. Two different models are shown: one that doesn't incorporate an iterative process and one that does. Since bitcoin prices follow a stochastic process, the paper utilizes an iterative ARIMA model to forecast single day price changes as opposed to the traditional approach to time series utilizing an 80:20 train test split.

Traditionally, finance and economics have used statistics and probability theory to gain insights into economic and business trends. The idea is that trends contain cyclical elements and are a composition of several time based factors (seasonality, frequency, general trend, etc). If companies, individuals, and governments can leverage the power of those factors and measure how those factors change over time, they can gain valuable insights and make better informed decisions. When examining changes over time, "the aim is to estimate how the sequence of observations will continue into the future."[1] For this study, those observations could lead to an increase in profits through arbitrage and long/short selling of bitcoin.

Simply put, time series analysis is a vital tool for forecasting changes over time and should be a cornerstone of every modern business. Time series has "tons of practical

applications including: weather forecasting, climate forecasting, economic forecasting, healthcare forecasting, engineering forecasting, finance forecasting, retail forecasting, business forecasting, environmental studies forecasting, social studies forecasting, and more."[2] It is for these reasons that our study's aim is to create a usable ARIMA model that can predict bitcoin prices given a MAPE of under 0.20.

Data collection was very straight forward as the study used publicly available daily closing prices from Yahoo's finance section, which includes many stock and asset tickers. The .CSV file provided by Yahoo finance was saved to the researcher's local computer and then read into the Python environment using Jupyter notebook. A few disadvantages of using the internet for data collection are reliability of data, security, and data formatting. Generally speaking, it is a good practice to ensure that the source of the data is reliable, which can be judged by examining online reviews, comparisons of alternate sources, and popularity among domain experts in the field.

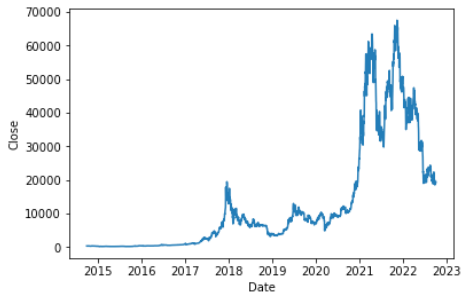
Yahoo! is a long time web service provider and has been a major internet business since their creation in the 1990s.[3] They have a host of services and several catalogs of information. There exist many sources of daily bitcoin closing prices and this study examined the data to ensure it correctly reflected data found in other financial sources. Another advantage is that this data was in a .CSV file which is the correct file type to read into the Python environment. Since we are considering daily bitcoin prices (and not smaller time steps like 4-hour candle closes), this data was the easiest to acquire.

All of the techniques and tools used for data preparation and extraction were chosen for their heavy use in modern industry and their availability as open source. Since Python and its related libraries, methods, extensions are open source, the study does not have any upfront costs other than using already purchased computer hardware. The advantage of using open source is that it is cheap and has extensive online documentation if problems arise. The disadvantage to using Python is that problems or errors in the code can take extra time to diagnose. Often times one must spend a few hours searching through documentation or online repositories in order to solve a domain specific problem. In this case, this study used many online resources to rectify problems or errors in the code which were mostly due to incorrect configurations or updates. However, the advantages of using free and open source software, despite sometimes running into temporary problems, outweigh the disadvantages.

The data was imported into a Python 3 Jupyter notebook using the Pandas read_csv function. The data is saved in the environment and then checked for nulls, missing, or incomplete observations. A trend line is visualized and summary statistics are examined. Typically, with time series data, it is appropriate to make the time step the index and the closing price the dependent variable. The screenshots below show the data cleaning process and calculations performed in order to clean and wrangle the data. Jupyter notebook was the IDE of choice for this study since it provides an easy to use format and is widely used in the industry of data science.

```
[5]: #Visualize as graph
     sns.lineplot(x='Date', y='Close', legend='full', data=df)
```

```
[5]: <AxesSubplot:xlabel='Date', ylabel='Close'>
```




```
[6]: df.drop(['Open', 'High', 'Low', 'Adj Close', 'Volume'], axis=1, inplace=True)
```

```
[7]: df.isnull().sum()
```

```
[7]: Close    1
     dtype: int64
```

```
[8]: df[df['Close'].isnull()].index.tolist()
```

```
[8]: [Timestamp('2022-09-28 00:00:00')]
```

```
[9]: df = df[0:2919]
     df.tail()
```

[9]:

Date	Close
2022-09-23	19297.638672
2022-09-24	18937.011719
2022-09-25	18802.097656
2022-09-26	19222.671875
2022-09-27	19110.546875

After the data has been cleaned it is checked for stationarity. Put simply, stationarity is achieved when a time series reflects a constant mean and variance. The image below visualizes stationarity and how changes in mean and variance affects time series.

There is a vast literature on stationarity and "several different notions of stationarity have been suggested in econometric literature over the years."[4] Due to the varying degrees on which stationarity may reveal itself, it may be more informative to think of a stationary process in the following way: the way in which a process changes *does not change*. Stationarity is therefore critical to finding an underlying structure to forecast upon. If our data is not stationary then it may not lead to a reliable forecast since the aim of forecasting is

to discover an underlying process not dependent on time; there needs to exist a cyclical process that underlies the time series in question.

Seasonality is another important concept to consider when studying any time series. It is helpful to view seasonality as a change in the degree of variance in a time series, often one that repeats on an annual basis. Seasonality has its epistemological roots in business cycle theory and econometrics. Seasonality can refer to annual changes or changes that occur on a different time step such as minutes or decades; seasonality refers to a cyclical process in the time series that affects the variance.

For bitcoin prices, it became very clear in the initial stages of this study that the bitcoin chart does not represent a typical time series and therefore a predictive model based solely on historical data, without context, was difficult to create. This problem is addressed by introducing a different approach using an iterative model update in its forecast.

After examining the data for stationarity using the Dickey-Fuller test and finding that the data was likely non-stationary, the next step was to find ideal parameters for p, d, and q. Since this study uses the pmdarima package, no manual differencing is needed. The auto_arima method finds the ideal p,d,q by trying different combinations of values until the lowest AIC score is found. An advantage of using auto_arima is that it will find the ideal parameters for our model by iterating through different terms and pick the best terms according to this metric with a lower score being better. Akaike Information Criterion (AIC) is a measure of model fit and is widely used in time series.[5] The hyper-parameters for both the primary and secondary models are shown below (they are the same).

SARIMAX Results

Dep. Variable:	y	**No. Observations:**	2335
Model:	SARIMAX(4, 2, 5)	**Log Likelihood**	-17358.540
Date:	Mon, 07 Nov 2022	**AIC**	34737.080
Time:	15:12:06	**BIC**	34794.629
Sample:	10-01-2014	**HQIC**	34758.047
	- 02-20-2021		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.5207	0.019	-27.043	0.000	-0.558	-0.483
ar.L2	-0.9931	0.021	-46.966	0.000	-1.035	-0.952
ar.L3	-0.4052	0.024	-16.984	0.000	-0.452	-0.358
ar.L4	-0.7402	0.021	-35.226	0.000	-0.781	-0.699
ma.L1	-0.4478	0.024	-18.449	0.000	-0.495	-0.400
ma.L2	0.4017	0.025	15.832	0.000	0.352	0.451
ma.L3	-0.4811	0.028	-17.348	0.000	-0.535	-0.427
ma.L4	0.2298	0.032	7.234	0.000	0.168	0.292
ma.L5	-0.5059	0.027	-18.832	0.000	-0.559	-0.453
sigma2	1.809e+05	1845.978	98.003	0.000	1.77e+05	1.85e+05

Ljung-Box (L1) (Q):	0.10	**Jarque-Bera (JB):**	191387.77
Prob(Q):	0.75	**Prob(JB):**	0.00
Heteroskedasticity (H):	2083.09	**Skew:**	0.85
Prob(H) (two-sided):	0.00	**Kurtosis:**	47.34

After the model parameters are found, pmdarima provides a forecast method where the prediction can be modeled and visualized. As can be seen below, the initial model's MAPE score is very high, suggesting poor goodness-of-fit. This is due to the train test split being too high as the model cannot account for a prediction that is as long as the testing data.

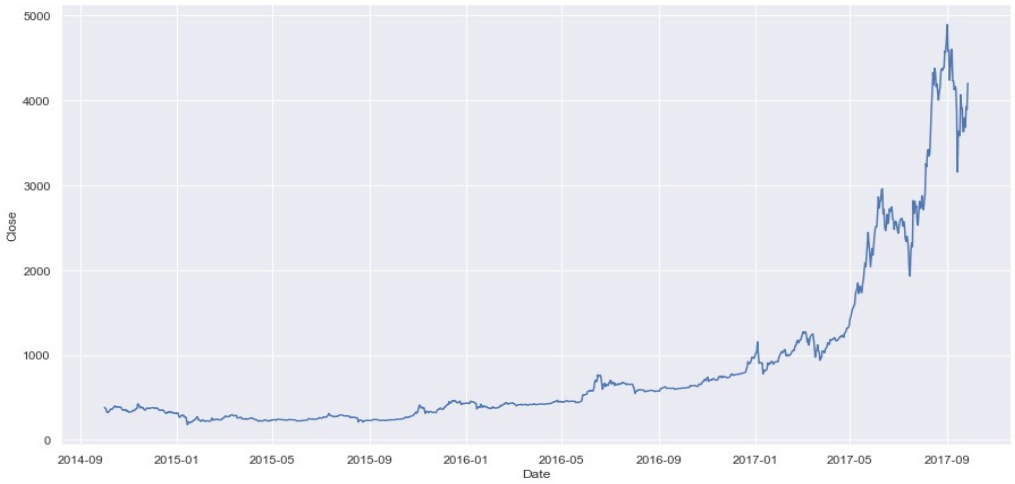
```
print(f"MAPE: {mean_absolute_percentage_error(testing, prediction)}")
```

```
MAPE: 8.142182995701603
```

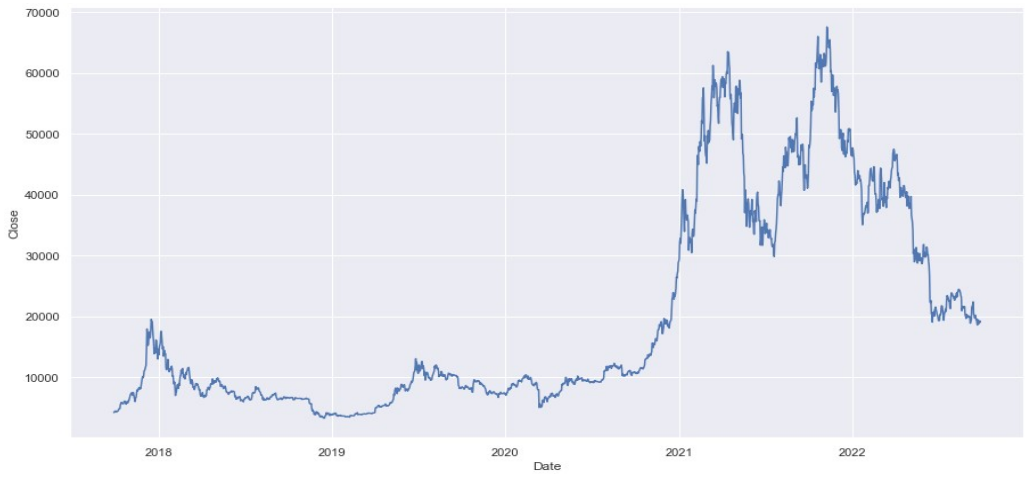
The above model using a standard 80:20 split with auto_arima deciding ideal hyper-parameters was not a good model for prediction since it had a MAPE of 814%. This is much higher than the hypothesis of under 20%. This is mainly due to the irregularity of bitcoin prices and the unpredictable nature of financial markets due to the Efficient Market Hypothesis which states that all prices contain all information.[6] Since prices contain all

information, they reflect that information *in the price,* which means that those looking to arbitrage on price by short or long selling will affect price by their strategy and therefore be unable to beat the market. In statistical terms, it means that seasonality and stationarity will be inconsistent between the training and testing set and will lead to poor prediction metrics. This study's aim was to create a *usable* prediction model utilizing the ARIMA methodology but since financial markets generally follow a random walk, there is no realistic way to use the standard approach. "All models are wrong but some are useful" is a quote by George Box[7], the creator of the ARIMA model. In the spirit of George Box, another approach is needed to make this model useful.

Since ARIMA will model the future based solely on the statistical properties of the training set, the training set itself must adhere to some of the principals of which ARIMA will operate. Mainly, that would be a stationary time series with seasonal/cyclical trends that repeat. It is apparent that the bitcoin price action from the first date of 2014-10-01 and the last date of 2022-09-29 has no obvious seasonality or cyclical trends. This is partly due to the range of prices changing dramatically throughout the training set. For example, the mean price of bitcoin up until September 27, 2017 was 837 USD vs 19,882 USD for the period after. Look at the chart for the first period below.



The chart above shows a parabolic rise in prices, suggesting a non-stationary mean that is rapidly increasing. Including these data points will lead to a model that will not generalize well, since the bitcoin price action after September 09, 2017 looks very different. A majority of the training data includes this parabola, so intuitively it would make sense that the final ARIMA model will have high errors. For reference, the figure below shows the data after September 09, 2017.



This is why the study's initial model performed so poorly; the cyclical elements that characterize a *predictable* time series are not present in the bitcoin chart (since the chart

represents a random walk). It is for this reason that an alternate model is introduced with the below screenshot showing a MAPE of 2.8%.

```
print(f"MAPE: {mean_absolute_percentage_error(y_test, forecasts)}")
MAPE: 0.028475596350937625
```

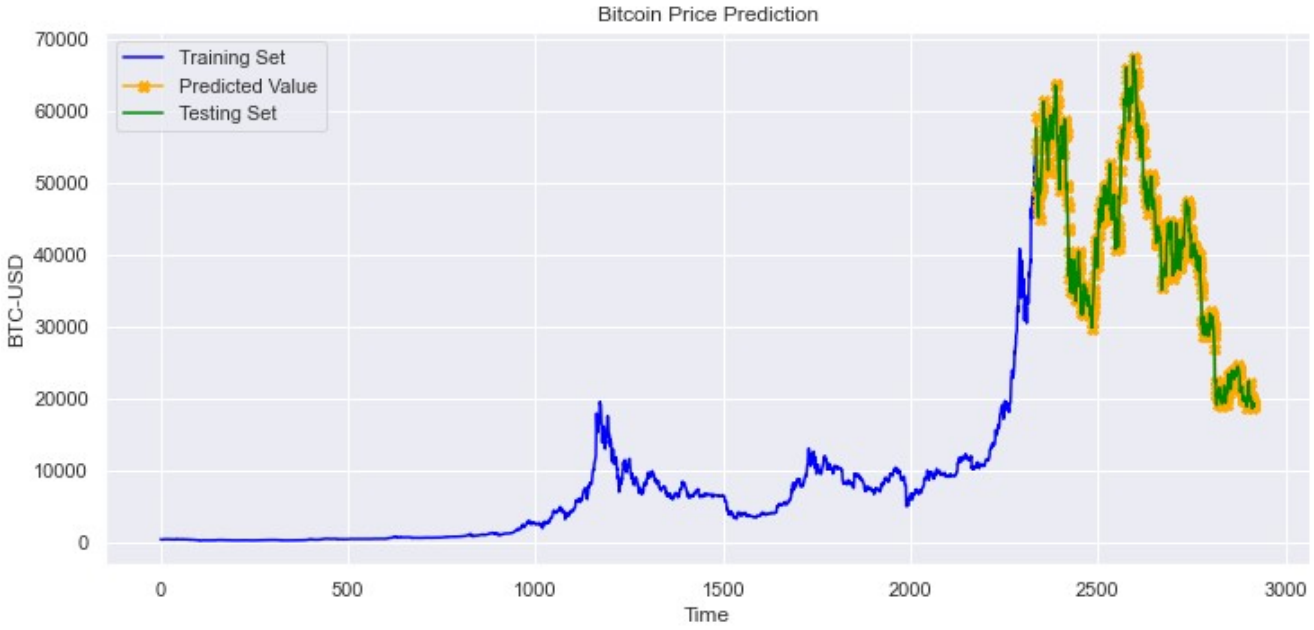
This new model showed a massive improvement over the original model with a MAPE well under 20%. This model architecture can be implemented into a trading strategy with a time prediction of 1 day and there is a significant chance that ARIMA can help investors and speculators increase their profit. The second model takes the prediction error of 1 day and then updates the training set and adjusts the model's prediction. Initially, the training and testing set are the standard 80:20 split but as the model updates, the prediction errors are saved (along with confidence intervals) and the model's predictions are updated with another day added to the training set. Using the ARIMA methodology in this way leads to a much better accuracy metric and provides a powerful tool for prediction tasks.

In conclusion, this study finds that a suitable model can be created by approaching the research question with usability always in mind and by utilizing auto_arima in the domain of stock market data. Financial modeling is very difficult since there is so much money to be made in the markets and it is for this reason that utilizing the power of machine learning and statistics is so important for investors and speculators. A limitation to this methodology is computation time as reiterating over model hyper-parameters can be time consuming even for a modern computer. The Jupyter notebook provided with this paper has reproducible code and can be independently verified by anyone who is curious about this study and its

findings. More information on this model can also be found in auto_arima's official online documents.[8]

Based on this analysis, the study recommends that the ARIMA model is a great tool to use for financial speculators and investors and with a Mean Absolute Percentage Error of just 2.8%, it has strong predictive ability. This study accepts the hypothesis that an ARIMA model can be made with an accuracy over 80% using a MAPE of under 0.20.

In conclusion, there are a few improvements that could be made to this model. Firstly, this study could improve by utilizing the model with a higher prediction horizon. Predicting 1 day into the future may be useful to some, but only those looking to take advantage of small price movements. Another improvement could be to the model hyper-parameters themselves by experimenting with different method argument values. For example, this model used a very simple approach to argument values and let the auto_arima function do almost all the heavy lifting. Another possible improvement could be to use a different unit-root test for configuring the differencing term. Overall, this model provides an excellent prediction tool for financial investors and speculators by utilizes a useful approach towards analyzing financial assets using machine learning. The image below shows the final model with orange showing the predictions and green showing the testing data.



Sources

1. Hyndman, R. J., Athanasopoulos, G. (2018). *Forecasting: Principals and Practice.* Otexts. Retrieved November 7, 2022 from https://otexts.com/fpp2/

2. Tableua (n.d.) *Time series forecasting: Definition, applications, and examples.* Tableau. Retrieved October 10, 2022, from https://www.tableau.com/learn/articles/time-series-forecasting

3. Wikipedia (n.d.) *Yahoo! Finance*. Wikipedia. Retrieved October 10, 2022, from https://en.wikipedia.org/wiki/Yahoo!_Finance

4. Palachy, S. (2019). *Stationarity in time series analysis.* Towards Data Science. Retrieved October 19, 2022, from https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322

5. Zajic, A. (2019). *Introduction to AIC – Akaike Information Criterion.* Towards Data Science. Retrieved October 19, 2022, from https://towardsdatascience.com/introduction-to-aic-akaike-information-criterion-9c9ba1c96ced

6. Downey, L. (2022). *Efficient Market Hypothesis: Definition and Critique.* Investopedia. Retrieved October 19, 2022, from https://www.investopedia.com/terms/e/efficientmarkethypothesis.asp

7. Wikipedia (n.d.) *All models are wrong*. Wikipedia. Retrieved October 19, 2022, from https://en.wikipedia.org/wiki/All_models_are_wrong

8. Smith, T.G. (2022). *Stock market prediction.* Alkaline-ml. Retrieved October 19, 2022, from https://alkaline-ml.com/pmdarima/usecases/stocks.html