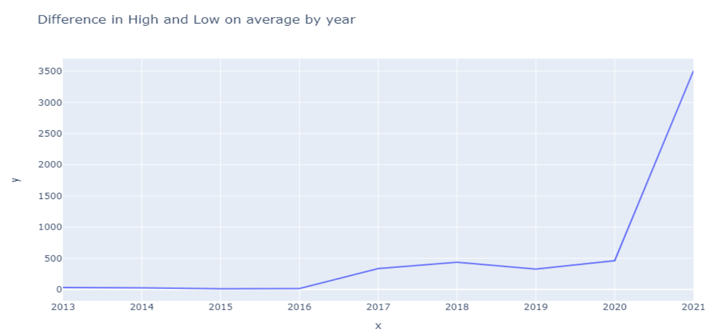
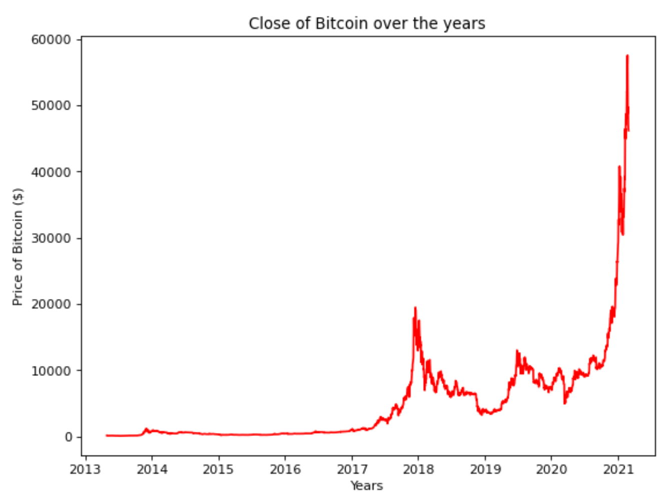
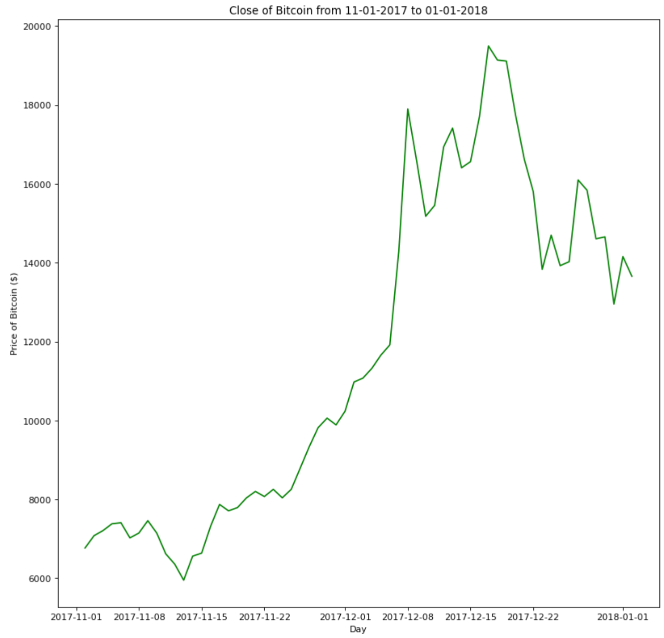
My case study report was to see if I could build a model that could predict Bitcoin's price in the future. This, in turn, could convince a business to invest in it.

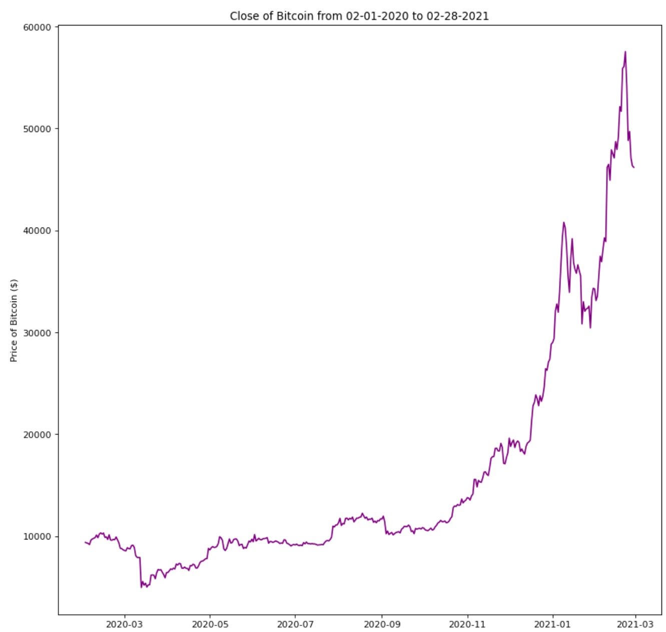
An introduction to Bitcoin-For hundreds of years, businesses have accepted payment for goods via fiat currency. But in 2009, an anonymous person created the first cryptocurrency called Bitcoin. This was deemed as a hedge to inflation, because of a max supply. Too, it is seen as a way for people to purchase goods and services that they otherwise couldn't due to a worthless currency. As Bitcoin became more popular, it has turned into an investment opportunity. Therefore, I want to convince stakeholders to buy-in by showing them that Bitcoin will continue to rise in value, thus giving them a great return on investment for the business. They should care about this because Bitcoin has a max supply thus it's great to hedge against inflation if they chose to hold onto it as a currency rather than sell it. Too, it isn't mainstream with the public and has the ability to get more investors, thus growing value, and giving out great return on investment.

As discussed here and Milestone 1, which I changed, is to try and convince the business to invest in Bitcoin. The data used for this dataset will come from Kaggle, specifically, https://www.kaggle.com/sudalairajkumar/cryptocurrencypricehistory?select=coin_Bitcoin.csv. This dataset has the high, low, open, and close price of Bitcoin from 2013 to 2021. It also has volume and market cap on those days.





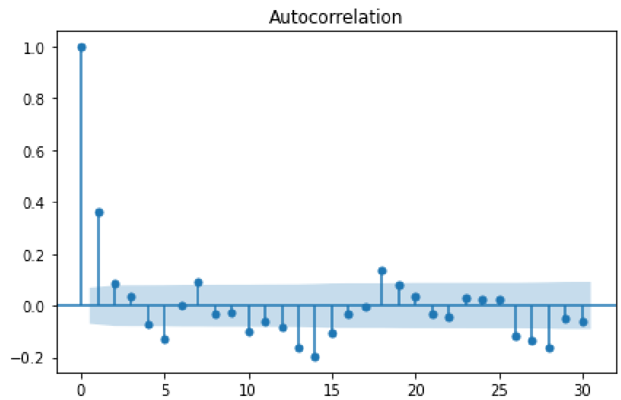


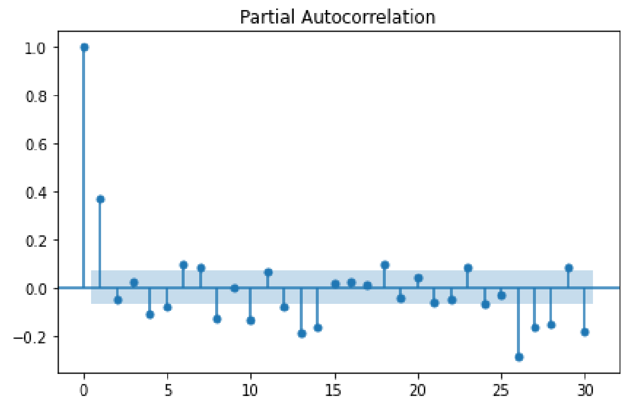


In Milestone 1, I showed these graphs which depict how much the value of Bitcoin has grown over the years and in short snippets of time.

In Milestone 2, I dropped 'SNo', 'Name','Symbol','Volume', 'Market Cap' since they aren't important to my analysis. Also, I added a column of the difference between the open and close prices. This was the extent of dimension and feature reduction.

In Milestone 3, I built my model and evaluated its performance. My model of choice was the ARIMA. This stands for Autoregressive integrated moving average. This model is fitted to time series data either to better understand the data or to predict future points in the series (forecasting). I chose this model because using the ARIMA model, you can forecast a time series using the series past values. First, I checked to see if the data had seasonality, which it didn’t. I did an autocorrelation and partial correlation plot. It shows the relationship between a variable's current value and its past values. An autocorrelation of +1 represents a perfect positive correlation, while an autocorrelation of negative 1 represents a perfect negative correlation.





Using these charts, I can finally build the model. An ARIMA model is characterized by 3 terms: p, d, q. Where,

p is the order of the AR term

d is the number of differencing required to make the time series stationary

q is the order of the MA term

First 5 lags is significant p=5. Since lag 1, 2, 3, 4 are near or out of the confidence interval, and lag 6 is in the blue area.

First order differencing makes the ts stationary, d = 1. An autocorrelation of lag 1 will measure the relationship between today's value (Yt) and yesterday’s value (Yt-1).

q = 9 since 9 are out of the blue in partial autocorrelation. This link shows general form of ARIMA model: https://ademos.people.uic.edu/Chapter23.html

	coef	std err	z	P>|z|	[0.025	0.975]
const	14.9308	10.919	1.367	0.172	-6.471	36.332
ar.L1.D.Close	0.1380	0.059	2.334	0.020	0.022	0.254
ar.L2.D.Close	1.1092	0.033	33.946	0.000	1.045	1.173
ar.L3.D.Close	-0.7016	0.071	-9.865	0.000	-0.841	-0.562
ar.L4.D.Close	-0.6466	0.035	-18.697	0.000	-0.714	-0.579
ar.L5.D.Close	0.4642	0.062	7.493	0.000	0.343	0.586
ma.L1.D.Close	-0.1026	0.058	-1.766	0.077	-0.217	0.011
ma.L2.D.Close	-1.1196	0.036	-31.334	0.000	-1.190	-1.050
ma.L3.D.Close	0.7365	0.073	10.041	0.000	0.593	0.880
ma.L4.D.Close	0.6370	0.047	13.635	0.000	0.545	0.729
ma.L5.D.Close	-0.4245	0.071	-5.969	0.000	-0.564	-0.285
ma.L6.D.Close	0.1162	0.034	3.450	0.001	0.050	0.182
ma.L7.D.Close	-0.1566	0.030	-5.230	0.000	-0.215	-0.098
ma.L8.D.Close	0.0157	0.021	0.733	0.464	-0.026	0.058
ma.L9.D.Close	0.2432	0.020	12.202	0.000	0.204	0.282

This is the code output of the ARIMA model. This shows the coefficients along with other useful information.

predicted=6143.054932, expected=6404.000000
predicted=6339.685964, expected=6385.819824
predicted=6311.531126, expected=6614.180176
predicted=6592.648083, expected=6529.589844
predicted=6578.897124, expected=6597.549805
predicted=6710.677158, expected=6639.140137
predicted=6709.212503, expected=6673.500000
predicted=6807.585096, expected=6856.930176
predicted=6954.381154, expected=6773.879883
predicted=6797.905858, expected=6741.750000
predicted=6810.109712, expected=6329.950195

My model is telling me that, based on a small snippet of predicted vs expected, that my model does a good job predicted its value

To conclude, my recommendations are that the business invest in Bitcoin either now or during a downturn for an investment opportunity. There is a lot of data here and thus takes Python a long time to run. But based on a small sample, it seems the model predicted vs expected is pretty close. I think if I had a faster processor on my laptop and more time I would be able to get a more accurate prediction.