Bitcoin Analysis:

The popularity of cryptocurrencies over the past few years has exploded. More and more people are investing their money into Bitcoin, Litecoin and Ethereum. What makes cryptocurrencies so appealing? Their popularity seems just as mysterious as their existence with new currencies appearing every day. With this mind, our group wanted to take a deeper look into Bitcoin and see if we could build a supervised regression model that would be able to predict the prices of Bitcoin and if there were any interesting trends that our model could return.

Proposal:

In this project, we attempt to conduct technical analysis using price trend and volume indicators. We rely on these variables, based on the assumption that they capture and reflect all public available information and hence focus on statistical analysis of price movements. The objective for our final project is to create a machine learning model to discover and take advantage of trends that can help us predict the future outcome of Bitcoin. Our dataset was obtained from Kaggle.com and it will feed our model over 3 million datapoints to learn from. Our work with this model and Bitcoin will become the foundation for future applications. Theoretically, our model can be applied to other cryptocurrencies and equities to help make savvy financial decisions.

The visualizations and dashboard for our project were created using JavaScript and the Plotly.js and D3 libraries. For the machine learning and analysis portion of our project, we used python and jupyter notebook. To view our work and analysis, you can run the included notebooks.

* bitcoin\_models.ipynb contains the our work to train and create our model along with a view different machine learning libraries that were used to see which gave us the best result. We ultimately used Scikit Learns built in model to perform our linear regression.
* Regression\_model.sav is the model that we created from bitcoin\_models.ipynb. We saved our model so that it could be loaded into other notebooks for analysis.
* Model\_testing.ipynb was created to run our model with current data on bitcoin. Our intentions were to use the model we created to predict other stock symbols based on the price and volume indicators.
* get\_data.ipynb was an additional notebook created to make API calls to Quandl and is not associated with the machine learning section of our project. The data obtained from Quandl and this notebook were saved and stored local for visualizations on our dashboard.

Data:

Before we talk about the data used for our machine learning model, I would like to briefly go over the data obtained and used for the visualizations for our dashboard. As stated above, the data was obtained using an API call to Quandl.com. The data was procured from Bitfinex which collects day to day data on various cryptocurrencies and is updated daily. Quandl allowed us, for free, to call the data on bitcoin for the period beginning April 2014 using their API. The indicators included in the dataset are: High, Low, Mid, Last, Bid and Ask price, also included was the volume traded for that day. This data was much smaller to work with and allowed us to create visualizations for our dashboard that we hope benefits the end user.

The data we used for our machine learning model was obtained from Kaggle. We initially started with the dataset obtained from Quandl, however, it became clear that the dataset did not include enough information to allow our model to learn and make accurate predictions. With a little digging, we were able to find a dataset that contained minute to minute data from December 2011 to March 31, 2021, yielding over 3.6 million rows of information to build and train our model. One issue that arised from a dataset this large was that it could not be shared amongst ourselves. In order to share the data, we stored it in our AWS S3 bucket and called the URL to our jupyter notebooks to perform the analysis. We did see some performance benefits by storing the dataset using this method. Now that we had our data, we needed to review and cleanse our data before progressing to the analysis. Our data preprocessing included checking NaN (Null) values (no Nulls found), formatting the Timestamp column from timestamp to Datetime and renaming some of the columns. You can see the final data frame with the cleansed data. This was plenty of data to start training our model. Lastly, our supervised model would take in 3 independent variables: high price, low price and volume. We created our final data-frame holding this data.

Model/Fit/Predict:

Now that we have prepared our data for analysis, we can begin to train our model so that it could make predictions. First, we need to define our independent and dependent variables. As indicated above, we are using a supervised learning model which means that we have data for what is expected from our model. We defined our independent variables as High price, Low price and Volume traded which will be held in the X variable. Our dependent variable, or the expected output from our model, will be the closing price stored in the y variable. Our next step is to split the data into a training dataset and a testing dataset. The reason we do this is so that the model can begin to learn from the training data before we test the accuracy of our model with the testing data. To remove any bias from the training and testing process, when the data is split, it is split randomly. Once this is complete, we can fit the data to our model and make our prediction. What you can see from our residual plot, which is a scatter plot that shows the dispersion of the data, is that our model’s predictions from the testing data is quite accurate compared to the training data. To provide a more concise visualization of our model’s prediction, we created another data-frame that shows the actual closing price versus the predicted price. In order to check the accuracy of our model, we fitted our data with several models (Linear regression, Lasso, Ridge and ElasticNet model) and selected the best model based on the r2 and mean squared error (MSE) scores. The selected model, Linear regression, had a r2 score and MSE of 0.9999 and 73.2997 respectively which produced the best predicted values. Now, let’s feed our model some current data and see what output our model returns.

Analysis:

Although our model can be tested on any stock, we chose to do a further testing on Bitcoin data from April 2021 to July 1, 2021, and we plotted the predicted results against the actual results. It is worth noting that both lines appeared as we had suspected based on the r2 and MSE values obtained from training the model. We then calculated moving averages from the actual closing values. The reason for calculating the moving average of a stock is to help smooth out the price data over the specified period by creating a constantly updated average price and this helps mitigate the impacts of random, short-term fluctuations on the stock price. We calculated a 15-day and 30-day simple moving averages (SMAs) as well as an exponential moving average (EMA). We chose to plot the 30-day moving average against the exponential moving average and they both depict a similar trend to the actual closing price and hence gives users a fair idea of the general price trend of the stock.