Investigating new features in Ethereum price prediction

Alexandra Duboy

Question: What are we trying to solve?

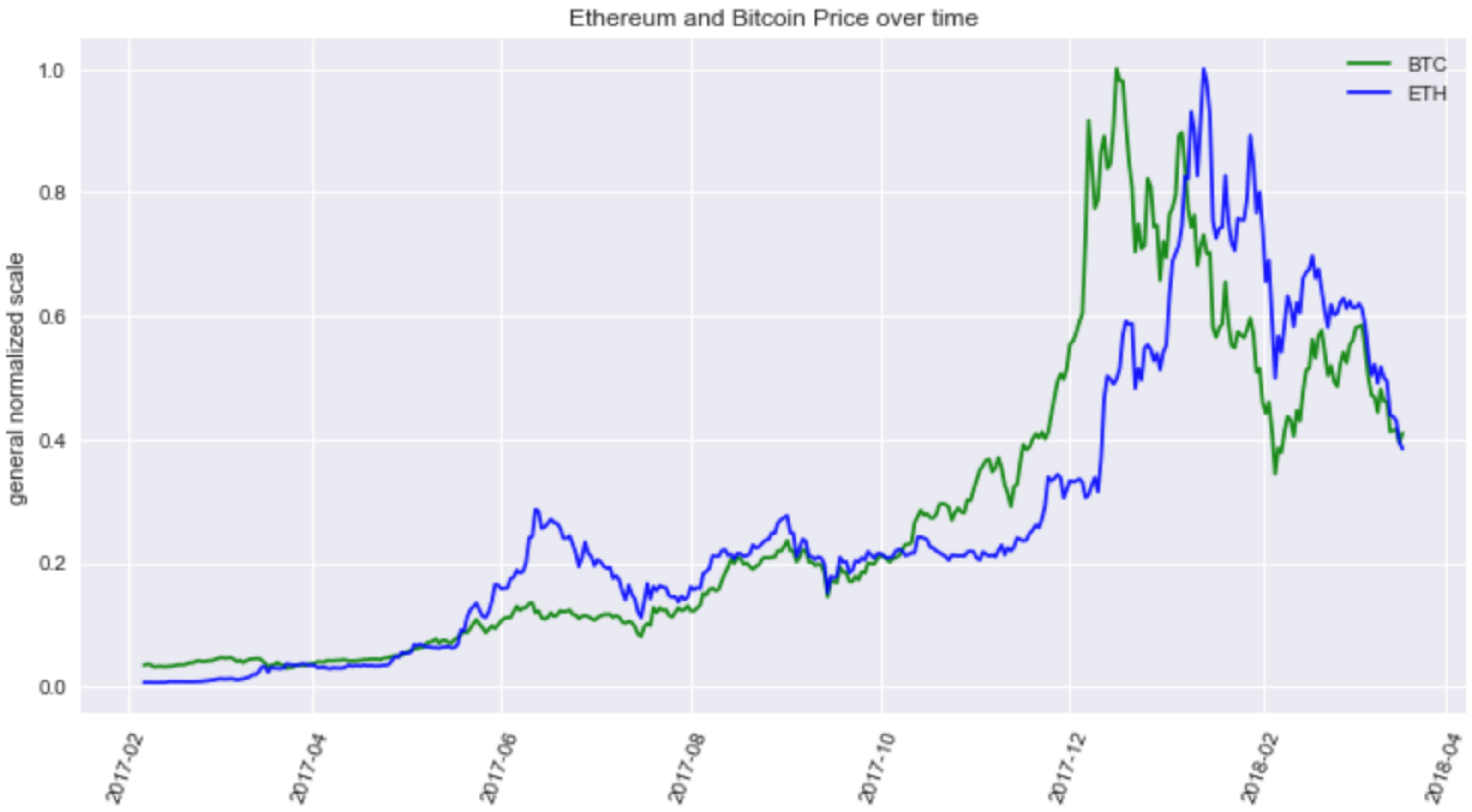
- We are asking the question what exactly affects ethereum price and what features can be investigated that can be fed into a predictive model?
- An Objective, or the impact of business value, would be the following: if we can see how something is correlated to ethereum price, we can make predictions about how the price of ethereum might respond when we see rises and dips in other quantities.
- Specifically I wanted to investigate the occurrence of a 3 gram of a [proper noun ,“backs”, “ripple”] or [proper noun ,“ban’s”, “cryptocurrency”] (the extraction of this utilizes tokenization and POS tagging)

Data wrangling:

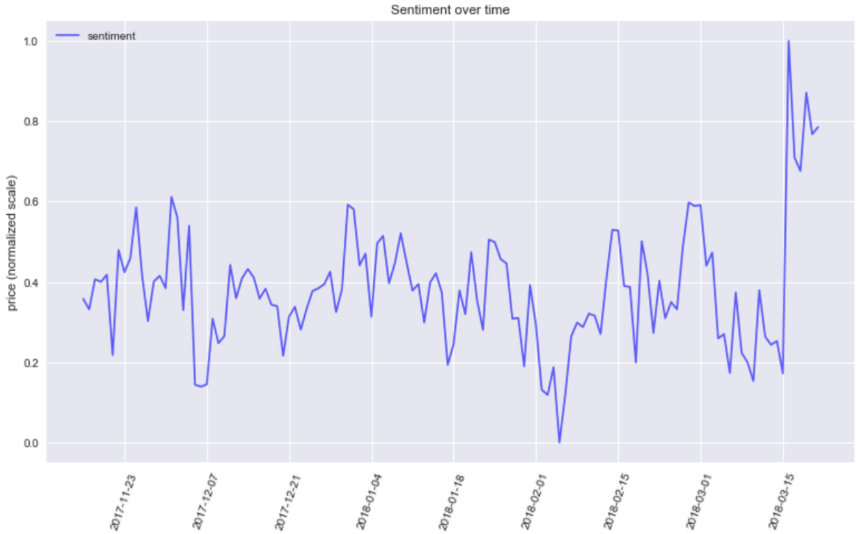
- To obtain the data we scraped using the reddit API, we went through about 15 subreddits related to cryptocurrency news and/or particular coins of interest. We performed overall sentiment analysis on the data we gathered and then we fed the “average sentiment per day” in as a feature into our model. We also fed in a count of an occurrence of n-grams that are specifically searching for news if a proper noun endorses or bans crypto, and seeing how that correlates with price.
- Google search frequency of certain terms used the PyTrends API
- Other data like Dow Jones Index daily values, and ethereum/bitcoin price daily values, were directly downloadable from the internet.

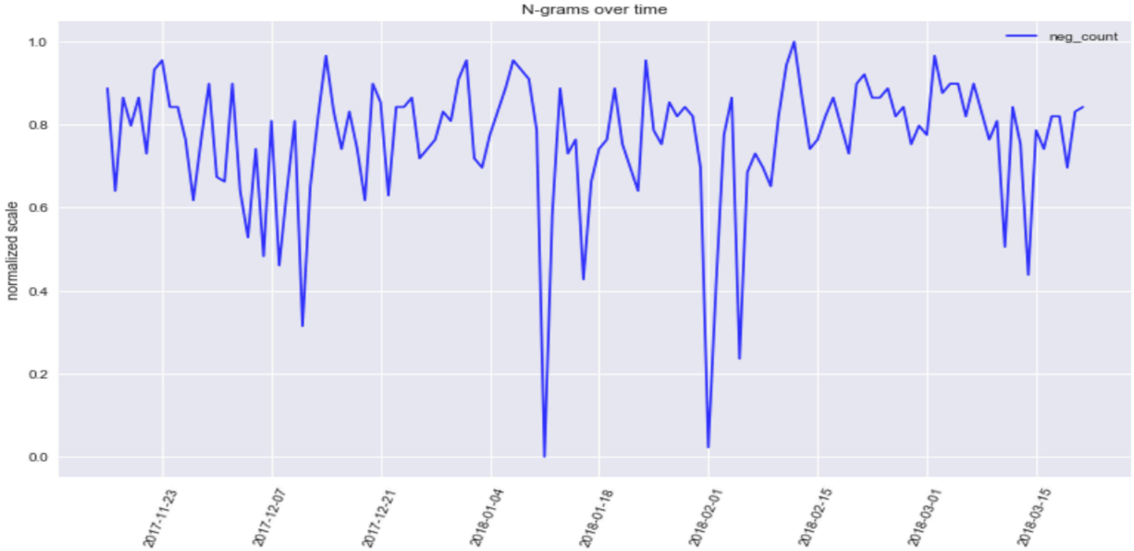
Now for some EDA of our data.

A graph of price over time of both Ethereum and Bitcoin.

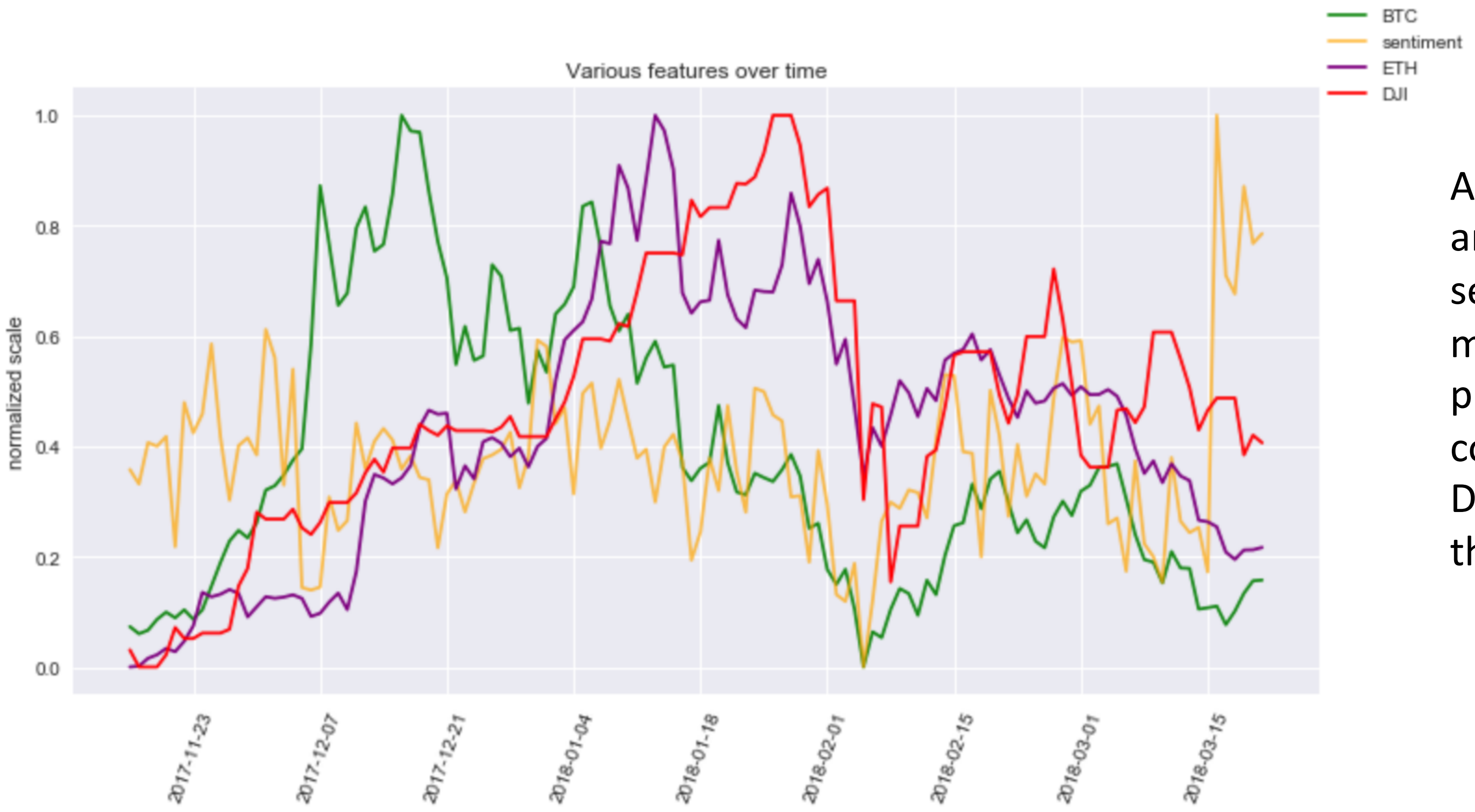


Sentiment over time and n-grams over time



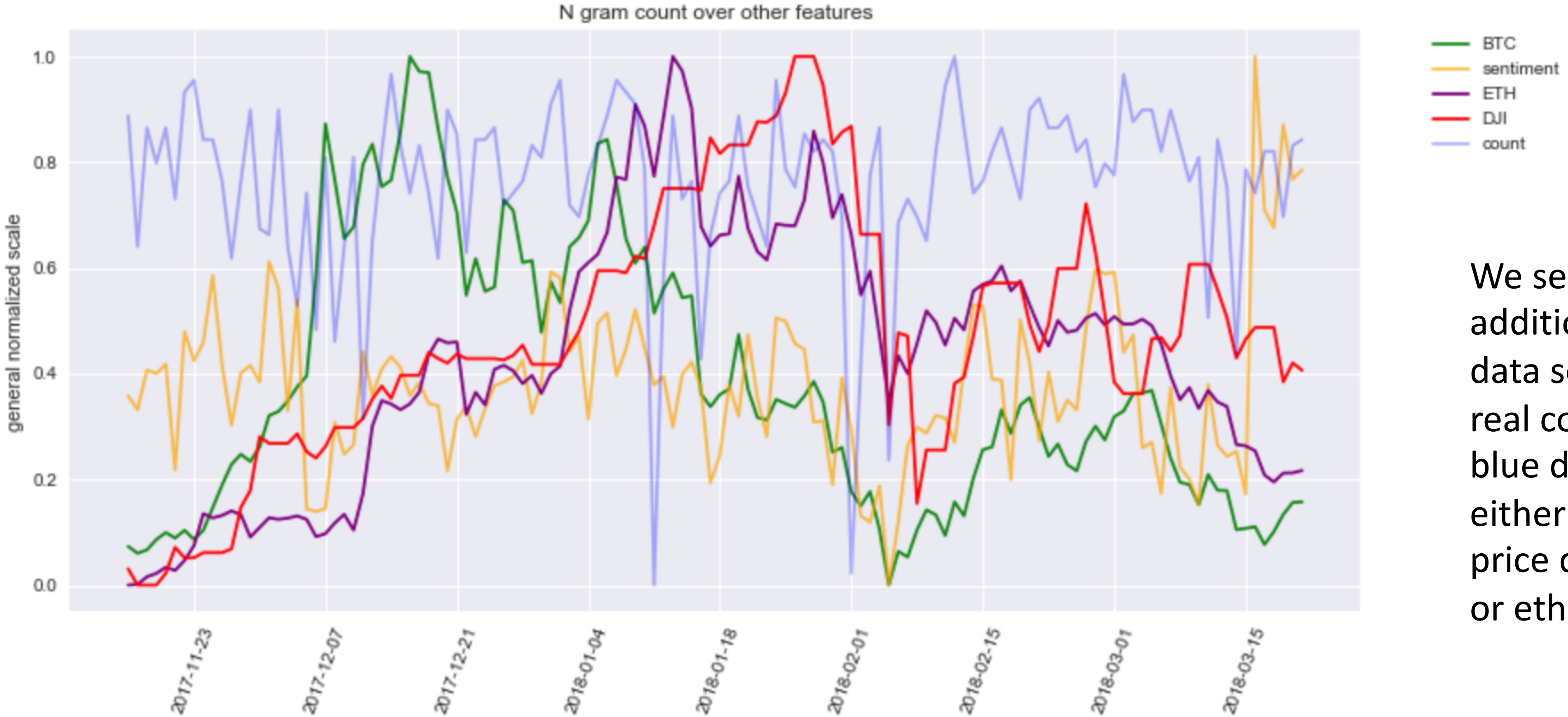


Multiple features over time



As you can see, there are two features that seem to over lap the most, the red and the purple lines, which correspond to the Dow jones data and the ethereum data

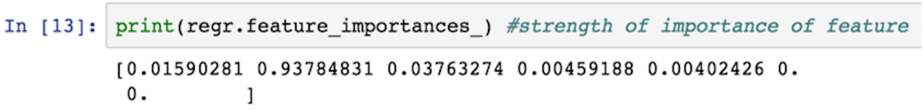
Multiple features over time (with “count” of n-grams added)



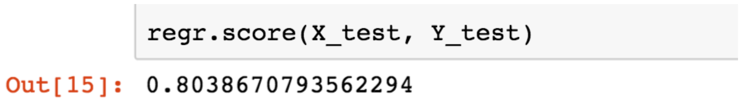
We see that the addition of the blue data series, there is no real correlation of the blue data series with either of the crypto price data, being bitcoin or ethereum.

Checking feature importance

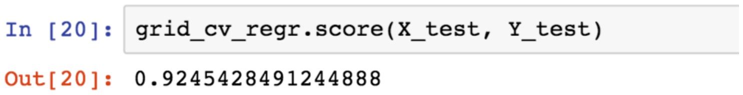
performing grid cv search, dropping some features to improve accuracy and performing grid cv search again.



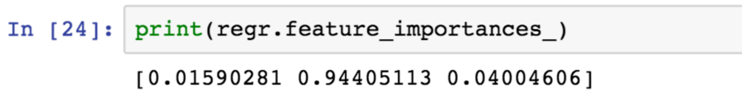
```
In [13]: print(regr.feature_importances_) #strength of importance of feature
         [0.01590281 0.93784831 0.03763274 0.00459188 0.00402426 0.
          0.         ]
```



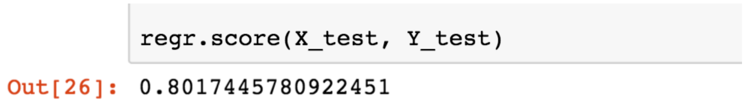
```
          regr.score(X_test, Y_test)
Out[15]:  0.8038670793562294
```



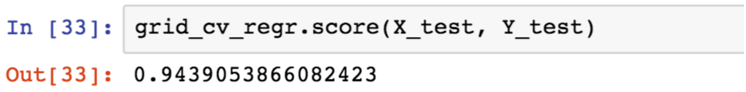
```
In [20]: grid_cv_regr.score(X_test, Y_test)
Out[20]: 0.9245428491244888
```



```
In [24]: print(regr.feature_importances_)
         [0.01590281 0.94405113 0.04004606]
```



```
          regr.score(X_test, Y_test)
Out[26]:  0.8017445780922451
```



```
In [33]: grid_cv_regr.score(X_test, Y_test)
Out[33]: 0.9439053866082423
```

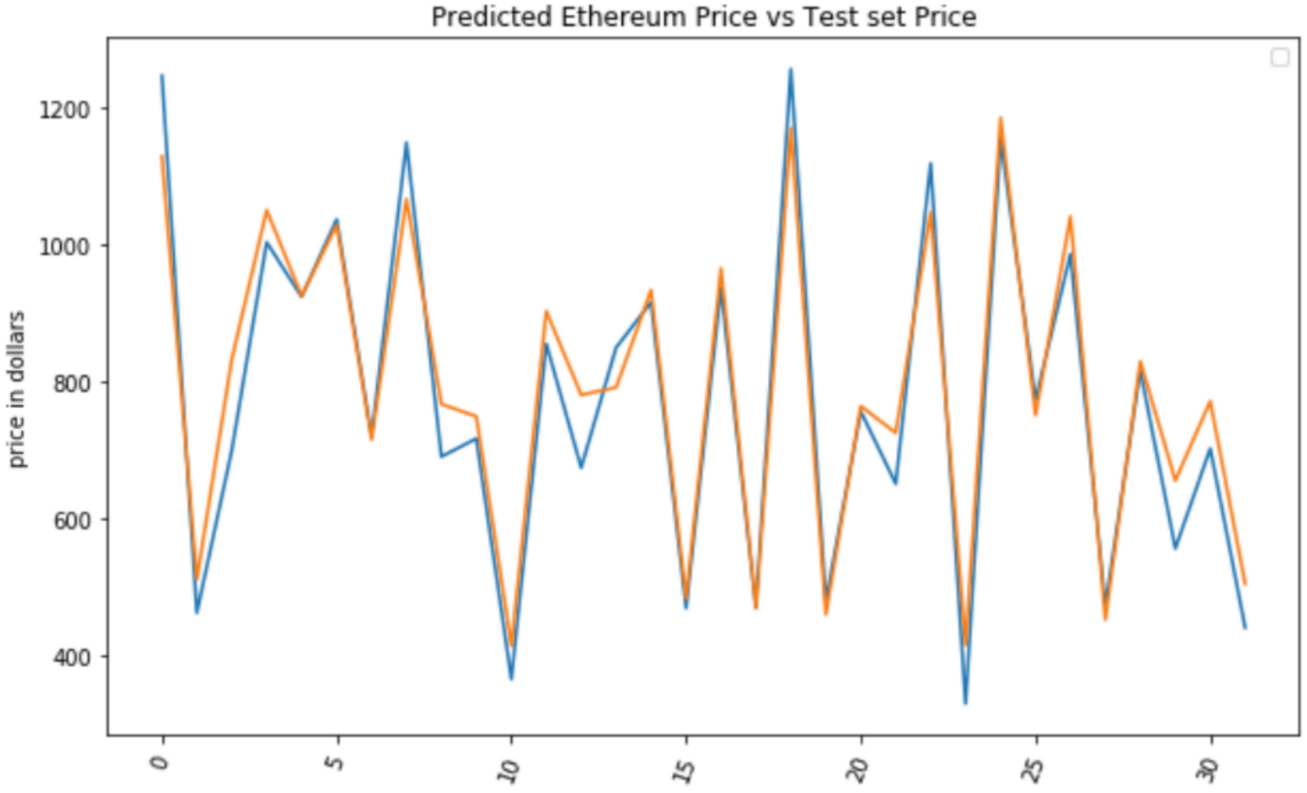
As you can see, a few features were not very strong, but notice that the second feature fed into the model was very strong. This feature corresponds to the dow jones index, the next strongest features ended up being bitcoin price and google search frequency but they were both very weak. Unfortunatley the sentiemnt and n-grams feature seemed to have a feature importance value of zero.

Modeling: Arima Time series prediction and Random Forest Regression

- First we tried Arima times series forecasting to try to predict price with some exogeneous variables.
- Why chose this model and why did it work? I looked towards this model because the arima time series model did not work out very well due to the large spike in the data. Time series forcasting only works on data that is of a certain type, one with relativley sinusiodal noise fluctuations and increasing fairly linearly. When splitting up the train and test data, the test data was all after the spike, so th etest data was all a big down slope.
- We then decided to try out Random forest, which worked very well with the DJI as the strongest feature. This, now, is not prediction, it is actually randomly shuffled, so its results are a predicted price given a certain Dow Jones index for a random date,
- So, based on how the model had been trained, when given a DowJones index number, this can predict with high accuracy, the ethereum price. Considering that the daily Dow Jones index is something that may be more easily predicted using ARIMA time Series, what can be done in the future is this: to conduct arima time series on the dow jones data, predicting the dow jones data, and then feeding the prediction of the dow jones as a feature into the random forest regressor. It would be a second order prediction but still may be a plausible technique.

Results from Random forest modeling:

The orange series is the predicted data points and the blue is the actual data points.



Conclusion.

- In Conclusion: the features extracted from scraping the reddit data were not very useful, though it was a great learning experience.
- what can be done in the future is this: to conduct arima time series on the dow jones data, predicting the dow jones data, and then feeding the prediction of the dow jones as a feature into the random forest regressor. It would be a second order prediction but still may be a plausible technique.