NLP Project – Prototyping

Jack Flitcroft

Nathan Chapman

5/3/2022

Our system

* Because our problem has nothing to do with predictions, or supervised learning, we don’t have any accuracy metrics or any statistical metrics.
* What we do have is a model (really, it is a dictionary of words with their associated clustered meaning) that can be used with a custom-built function to cluster tweets about bitcoin based on sentiment. Basically, with a bunch of tweets, we can calculate sentiment for positive, neutral, or negative for each one, and then graph the general sentiment of the set of tweets to see the current ‘vibe’ of twitter surrounding a given subject.
* How it works:
  + First, we download a bunch of tweets, for us we took the last 15 days to use.
  + Then we drop duplicates, grab only alphabetic characters, remove stop words, lemmatize the words, concatenate them, get rid of non-English tweets, get rid of tweets with less than 10 likes, and get rid of tweets that do not have the word ‘bitcoin’ in them.
  + Once the data is processed, we saved the first 2000 of 8443 tweets to use as a test set. The tweets are still ordered by date, so these will represent the last couple of days out of the whole set, making it the most current sentiment.
  + On the rest of the tweets, we used word2vec to train over 80 epochs, so we could represent the whole of the tweets in the training set as a bunch of vectors.
  + Then, we used kmeans clustering to cluster the vectors into 3 groups. Ideally here they would cluster on positive, negative, and neutral sentiment, but you will see what happened.
  + We then labeled the tweets with a cluster value of -1, 0, or 1, based on the cluster they are in. -1 is supposed to be negative, 0 neutral, and 1 positive. This was done by looking at the most prominent words in each cluster and trying to manually figure out which one was mostly positive, most negative, etc.
  + There were definitly some outliers, so we manually changed those to be pos, neg, and neu.
  + Finally, we plotted a pie chart where it was broken down into the number of words in each sentiment. This was visually allowing us to see how large each group was.
  + Then we saved the vocabulary and each words sentiment as a dictionary, this is essentially our model.
  + Using this dictionary, we can compute the pos, neu, or neg score for each tweet through a custom-built function which you are going to run over the data. This is what uses our ‘model’ to evaluate the data.
* How you ‘run’ our model:
  + The jupyter notebook can be run in order. Run the first two cells to load dependencies and the sentiment function.
  + Then load the next cell which will read in the model, the test data, and compute the sentiment on each tweet in the test data.
  + Run the following cells to find the distribution of tweets based on sentiment, and the word clouds for each group. This is a visualization of the sentiment in the test data, and what each group looks like.
* Our model is not meant to be a predictor, but to gauge a general sentiment of the current community on twitter talking about bitcoin. That is why the only metric is how well the groupings worked, and if the sentiment makes sense given the current price fluctuations of the market.
* However, our model clustered it into 3 distinct groups: what seemed to be normal crypto talk, people screaming “TO THE MOON”, and bots…

Some example output for bots group:

('jermehlovesbtc', 0.9906570911407471),

('btcphteve', 0.9815545082092285),

('crbear', 0.9799525141716003),

('bitcoindragon', 0.9793529510498047),

('bitchungus', 0.9792702794075012),

('daddybtcpleb', 0.9786920547485352),

('btcplumber', 0.9780187606811523),

('albertahodl', 0.9780045747756958),

('micajahautrybtc', 0.9728124737739563),

('denverbitcoin', 0.9726688265800476),

('sleepypubba', 0.9703138470649719),

('yooperhodl', 0.9689769744873047),

('knowun', 0.9687477946281433),

('ottohodl', 0.9609148502349854),

('bigseanharris', 0.9605709910392761),

('bitcoinbrabant', 0.9581685662269592),

('neiljacobs', 0.9554318785667419),

('hodltarantula', 0.955389678478241),

('btcliotta', 0.9544211030006409),

Example for reddit people (people yelling TO THE MOON):

('normies', 0.9955633878707886),

('gl', 0.9949708580970764),

('btccharlie', 0.9948376417160034),

('cryptobahamas', 0.9936908483505249),

('beautyofhelin', 0.9931666254997253),

('bitcoincrypto', 0.9916816353797913),

('intocryptoverse', 0.9909007549285889),

('specifically', 0.9906155467033386),

('tie', 0.9905450344085693),

('goingparabolic', 0.9903391003608704),

('infinity', 0.9902291297912598),

('diamond', 0.9902241826057434),

Finally, output for legitiment crypto talk:

('dex', 0.9831305146217346),

('bnbchain', 0.9801010489463806),

('etherum', 0.980091392993927),

('wonderful', 0.9764628410339355),

('r', 0.9759265184402466),

('ico', 0.9720548391342163),

('whatsapp', 0.9716647863388062),

('everyones', 0.9715234041213989),

('casino', 0.971429705619812),

('glorious', 0.9707171320915222),

('hodler', 0.9702656865119934),

('rock', 0.9698460698127747),

('minting', 0.9694404006004333),

('axionnetwork', 0.9691951274871826)

So obviously its hard to really control how the clusters get formed. We wanted to see 3 groups of tweets that were based on market sentiment, but it seems that for us to make that happen we will have to manually move positive, negative, and neutral words onto the sheet and maybe re-run the k-means clustering then? I am not sure, but the results sure are interesting to begin with!

We can definitly see there is a difference between the groups of tweets. Another interesting thing is the distribution of words in each group which you can see below:

Chart, pie chart

Description automatically generated