Steemit Post Text Analysis and Price Prediction

UNIVERSITÀ DEGLI STUDI DI MILANO

LA STATALE

Deparment of Economics,Management
and Quantitative Methods
Data Science and Economics

Project of Text and Sentiment Analysis

Murat Aydın

May 2021

Abstract. Topic modelling is a statistical unsupervised technique for discovering abstract topics that occur in a collection of documents. This paper presents a topic modelling for the social media posts of Steemit. Steemit is a block-chain based blogging and social media website where users can gain a cryptocurrency for publishing and curating content. The research question of this analysis was to find possible correlations between topics and price of Steem cryptocurrency. For topic modelling, we used LDAmallet statistical model from python gensim library. Coherence measure is used to decide about the number of possible topics. Given the randomness of the method used, 12-14 topics turned out to have the best results in this analysis where best is coherence value. To calculate the correlation between Steem price and Topics, we used growth rate. For steem price, we calculated daily growth rate through close price while for topics, we calculated growth rate with respect to their usage in a given day. This way, it was possible to see what topics are spoken more or less with respect to the price movements in a given period. Among all the topics considered, four topics were chosen to be used in price prediction of Steem cryptocurrency. They were posts about Steemit, Bitcoin-Cryptocurrency, Market-Finance and posts related to the rewards given to the users. After assigning the topic class given by the LDA algorithm to each post, we calculated daily polarity result through a pre-trained model trained on 1 millions of tweets for each day and topic. The resulting matrix then given to a LSTM model for final price prediction of Steem price. The LSTM model with textual information out-performed the one without textual information.

1 Introduction

Topic modelling is an unsupervised approach used for finding bunch of words called "topics" in large cluster of texts. Topics can be defined as a repeating pattern of terms in a corpus. Therefore a good topic model should result in meaningful words for a given topic. If for example our topic is about bitcoin, then we should be able to see words like "bitcoin", "market", "coin", "cyrpto","blockchain", "buy".

Topic modelling might be highly relevant in the case of Steemit since it has its own social media and cyrptocurrency where textual posts of users in this social media might have an effect on the Steem

cyrpto-coin.

2 Research question and Methodoloy

In this project, we will try to come up with topics where Steem-Posts cluster in and see if we can use such information to predict its price. Therefore, the research question of this project is to see if there is a relationship between textual posts published in steemit and its price.

To reach such aim, we need used pearson correlation in order to find relevant topics for the Steem price. For each day considered in the analysis, we calculated the growth rate of Steemit currency through close price and growth rate of a given topic's usage.

	Daily_growth	topic_growth	TimeStamp
0	0.043813	0.002096	2017-12-01
1	0.111111	-0.060057	2017-12-02
2	0.016667	0.173359	2017-12-03
3	0.172131	-0.052027	2017-12-04
4	0.020979	-0.070409	2017-12-05
5	-0.123288	-0.017065	2017-12-06
6	0.601562	0.106073	2017-12-07
7	-0.053659	0.052140	2017-12-08
8	-0.139175	-0.188095	2017-12-09
9	0.000000	0.084594	2017-12-10

Figure 1: Python Output

An example is provided in Figure 1 where we see Steem-Daily price growth calculated through close price and daily topic growth of a given topic, say bitcoin related topics, with respect to the previous day. The Daily usage growth of a given topic is calculated as follows:

Let Ca_T = total count of topic A at time t

and let CX_T = total count of all topics at time t

$$Ra_T = \frac{Ca_T}{CX_T} \tag{1}$$

Where Ra_T is the relative usage of Topic A at time T. Given Ra_T and $RA_T - 1$ we can calculate the growth rate of topic A's usage for any given day which is given by :

$$Ga_T = \frac{Ra_T - Ra_{T-1}}{Ra_{T-1}} \tag{2}$$

Here the Ga_T represents the values under the *topicgrowth* column in figure 1. This way we can easily see if a topics usage is correlated with the steem price growth.

However, there is a problem with this method since users might always talk about certain topic when the steem price goes either way. Imagine, in a given day Steem price had a wild movement and went up. We would expect topics about bitcoin or cryptocurrencies or financial markets to be spoken more on that day. And Imagine, the next day steem had another wild movement and it crashed. What would we expect? We would still expect such topics to be spoken more because people will speak about it more both when it goes up and crashes. Therefore, when we look for correlation between steem price and topic daily growths, we cannot keep the time range too long otherwise the correlation will converge to zero. For this reason, we calculated the correlations both monthly and arbitrary time range. If we keep the time range monthly or longer, since topic usage will grow any time when steem price climbs the peak or crashes, the correlation that we have in shorter periods cancel out and it converges to 0.

2.1 Dataset

The dataset is provided by computer science department of University of Milano, therefore it is not publicly available. Given the huge number of posts published each month, it was way over our computational power to process all the data. On average, a given month had around 7 millions of posts. This not only prevented us to use all the data but it also prevented us to do the analysis over a long period of time. That is why we focused on monthly analysis. We relied on random sampling. This raised another problem because since we are doing topic modelling, we should not distract the rel-

ative daily usage of posts during the sampling otherwise the topics would not be representative of the original data. To solve this problem, while randomly sampling the posts, we kept their frequencies consistent meaning that if on day1 total number of posts represents say, %5, of the total dataset, in randomly sampled dataset day1 still represented %5 of the total randomly sampled dataset so that topic daily growths would have almost identical result if we had done the analysis on the whole original dataset.

The results shown in this project is posts extracted between $2017-12-01$ and $2017-12-31$ and total number of posts sampled from the original dataset is 100.000

2.2 Data Pre-Processing

Given the nature of posts (some of them contained html codes), it was highly complicated to clean them and it is done with utmost attention. The whole text is lowered, cleaned from square brackets, punctuation, numbers, links, html codes, emails, newline characters, single quotes, emojis, urls, tags and some other complications raised by the html codes.

After carefully cleaning the text, we tried to understand the language that the text is written in. This took lot's of time since mainstream language detection libraries do not recognize some languages like thai. After language detection, we only focused on posts that are written in english.

Next, using gensim, spacy and nltk libraries, we cleaned the text from stop-words, created bigrams and lemmatized the text keeping only nouns, adjectives, verbs and adverbs.

After this step the topic modelling algorithm, LDAmallet of Gensim library, was ready to be fed.

3 Polarity Calculation

For polarity calculation, we trained an LSTM model provided in Kaggle[1]. This model is trained on 1 millions of tweets which have positive and negative classes. Polarity score is calculated for each topic and each day which is then averaged to get a single value for each topic and each day. Using this model provided us a big advantage. Since the Steem-posts do not have any label assigned, we could not be sure of the polarity results if we directly trained the model on Steem-posts. Training the model on a labelled dataset increases the confidence on polarity calculation.

4 Experimental Results

In figure 2, we present the label distributions of tweets. As we can see, it is a highly balanced dataset.

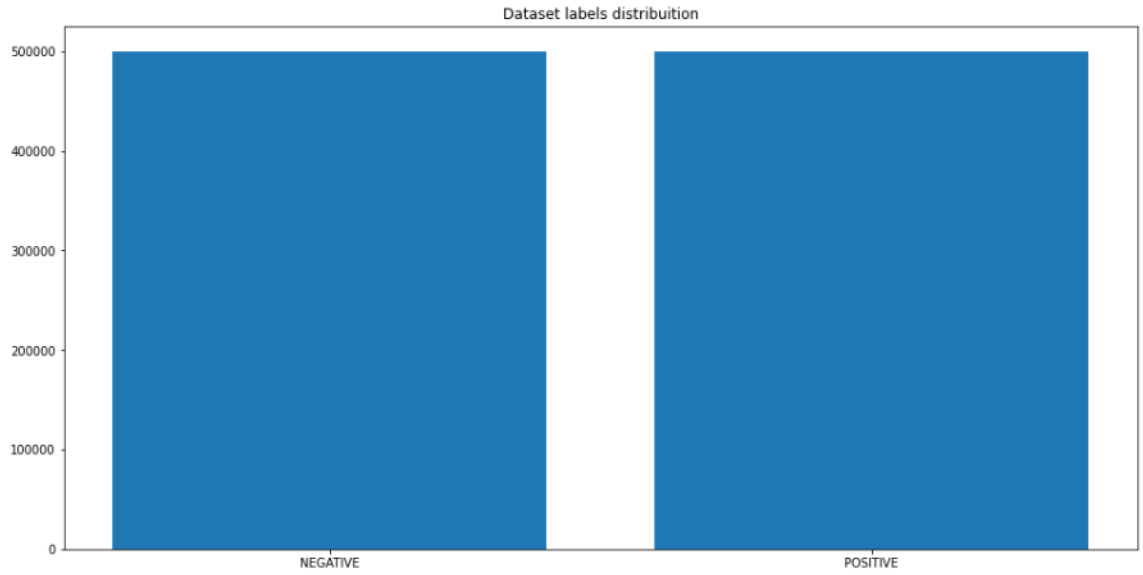

Figure 2: Python Output : Tweet label distribution

Balanced datasets provide higher results on accuracy metrics like Recall and F1 score.

	precision	recall	f1-score	support
NEGATIVE	0.79	0.77	0.78	99611
POSITIVE	0.78	0.80	0.79	100389
accuracy			0.79	200000
macro avg	0.79	0.79	0.79	200000
weighted avg	0.79	0.79	0.79	200000

Figure 3: Python Output : Tweet sentiment classification confusion matrix

Figure 3 presents the confusion matrix of the model. The accuracy turned out to be %79 and the other metrics give a promising result.

We used coherence metric provided by LDA algorithm to decide about number of topics selected. Choosing a high number of topics was problematic for two reasons.Recall that we are using about only %1.5 of the whole dataset. Firstly, since we sampled the original dataset and significantly reduced the number of posts, having a high number topics with a small dataset causes problems when calculating the topic growth rates. Imagine we have 100.000 posts like in this case, after cleaning the text and extracting only the english ones, assume the dataset is reduced to 50.000. So on average there is around 1.600 posts per day. If we have a high number of topics say, 20, calculating growth rates will not be consistent among 1.600 samples daily. Also, some topics might not be spoken at all in a given day since the daily sample size highly small. This will lead to NaN during the growth rate calculation.

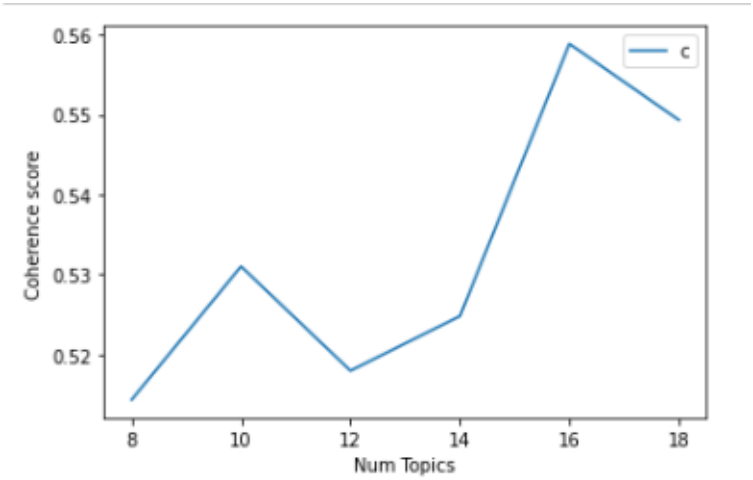

Figure 4: Python Output : coherence values

Given the above consideration, it was critical to decide about number of topics. Figure 4 shows the coherence values provided by the algorithm. The difference between coherence values is not that significant. Therefore there is no any reason to choose 16 which would make the model too complicated because of the problems described above.10 created overlaps between topics while 14 provided a good trade-off.

```
{0: ['create, make, user, work, account, video, add, platform, open, wallet'],
 1: ['year, world, country, music, place, city, people, movie, call, travel'],
 2: ['day, center, brbpost_create, photo, centera, image, payout, picture, btotal_upvote, challenge'],
 3: ['man, feel, love, time, day, child, back, home, eye, hand'],
 4: ['post, receive, vote, link, send, sbd, time, photo, follow, share'],
 5: ['post, nice, share, follow, great, work, give, good, content, write'],
 6: ['people, make, life, thing, time, person, work, world, mind, word'],
 7: ['make, food, water, beautiful, eat, tree, color, add, small, flower'],
 8: ['upvote, post, steemit, comment, information, reward, view, click, reply, number'],
 9: ['steemit, love, friend, year, good, find, community, hope, project, happy'],
 10: ['body, high, form, energy, find, human, level, result, increase, time'],
 11: ['game, time, play, make, good, start, thing, back, watch, point'],
 12: ['bitcoin, full_story, source, br, price, coin, hr, hr_classpullleft, service, buy'],
 13: ['market, money, company, report, price, business, blockchain, currency, system, transaction']}
```

Figure 5: Python Output: Dictionary representation of Topics.

Figure 5 shows the keywords that each topic has.The key of the dictionary is the topic number while the values is the keywords. We can see that there is actually 3 cluster here. Topics 12-13 being financial, 4-5-8-9 being about Steemit and the rest is about daily life of users. After getting the topics, we created a tabular structure and assigned a topic number to each post like shown in figure 6.

	Dominant_Topic	Keywords	Text
0	7	make, food, water, beautiful, eat, tree, color...	more americans have died at the hands white ch...
1	1	year, world, country, music, place, city, peop...	wolfpack ninja tour in colorado im actually on...
2	9	steemit, love, friend, year, good, find, commu...	thanks a lot
3	4	post, receive, vote, link, send, sbd, time, ph...	has voted on behalf of minnowpond if you would...
4	7	make, food, water, beautiful, eat, tree, color...	great shot gee the sunrise is beautiful where ...
...	...	...	...
65246	6	people, make, life, thing, time, person, work,...	hurry its the long awaited and everyone is dan...
65247	3	man, feel, love, time, day, child, back, home,...	is that a lamp in the right hand second story ...
65248	12	bitcoin, full_story, source, br, price, coin, ...	that's correct as a matter of fact no investme...
65249	0	create, make, user, work, account, video, add,...	thank you
65250	9	steemit, love, friend, year, good, find, commu...	yay luck my friend and i wish you the best in ...

Figure 6: Python Output : Tabular Structure of Topics assigned

Given the tabular form in figure 6, it was straightforward to calculate the topic daily growths as shown in figure 1 after joining this table with the time-stamp of each post.

Topics	Pearson_corr	P_val
0	0.203896	0.271245
1	-0.139216	0.455114
2	0.554160	0.001218
3	-0.400473	0.025583
4	-0.288830	0.115061
5	0.120579	0.518195
6	-0.411012	0.021622
7	-0.229594	0.214062
8	-0.090698	0.627509
9	0.281324	0.125243
10	-0.150393	0.419350
11	-0.232807	0.207542
12	0.107822	0.563706
13	0.111272	0.551222

Figure 7: Python Output : Monthly Correlation Analysis

In figure 7, we present the correlation results between $2017-12-01$ and $2017-12-31$. Here the topic numbers are the same as presented in figure 5 so a more intuitive interpretation can be achieved by matching them with the keywords in the dictionary presentation.

Above we already explained the reasoning of why a monthly correlation is not reasonable. Therefore, we do not think that this table is informative enough. However, we can still see that some topics are positively correlated with the price and some of them are even statistically significant. These are topic 2,3 and 6. With a quick look on the dictionary in figure 5, we see that they are social topics about day, love, child, home, life, people. We therefore conclude that this is a spurious relationship.

We will now present some of the correlation results within a smaller period. Given high number of topics, it is not feasible to present all.

Topics	Pearson_corr	P_val
0	0.514076	0.156829
1	-0.345902	0.361865
2	0.716983	0.029712
3	-0.811938	0.007860
4	-0.372524	0.323474
5	-0.184424	0.634781
6	-0.682978	0.042593
7	-0.574741	0.105492
8	0.370581	0.326204
9	0.386555	0.304099
10	-0.446971	0.227735
11	-0.134922	0.729275
12	0.580772	0.101041
13	0.140201	0.719021

Figure 8: Python Output: Correlation Analysis between 7 and 15 December

Figure 8 presents a weekly correlation result. Topic 2 is still highly correlated and statistically significant. Bitcoin related topics is number 12, financial market related topics is number 13. 8 and 9 are about the topics related to the structure of the Steemit. Given the reward-incentive structure of Steemit, we think these two topics could be related to the price.

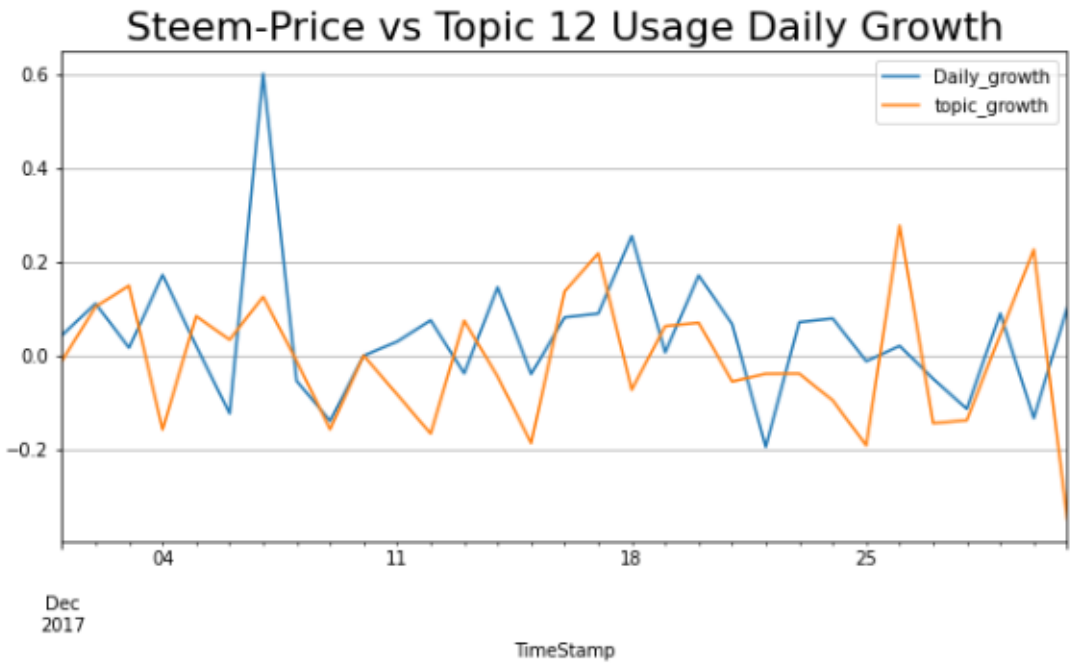

Figure 9: Python Output: Bitcoin related topics vs Steem Price growth

While in figure 9, we show the monthly series of steem price and bitcoin related topics growth.

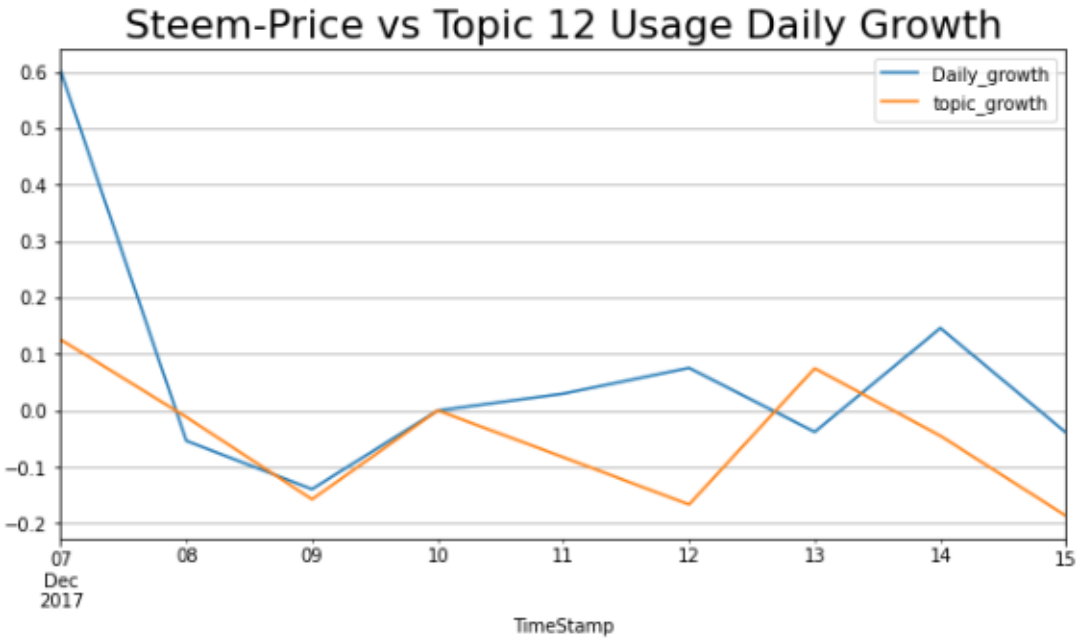

Figure 10: Python Output: Weekly Bitcoin related topics vs Steem Price growth

Figure 10 shows weekly correlations. Even if in this week, bitcoin related topics was highly correlated, around %58, it is not statistically significant. The p value is slightly over %10.

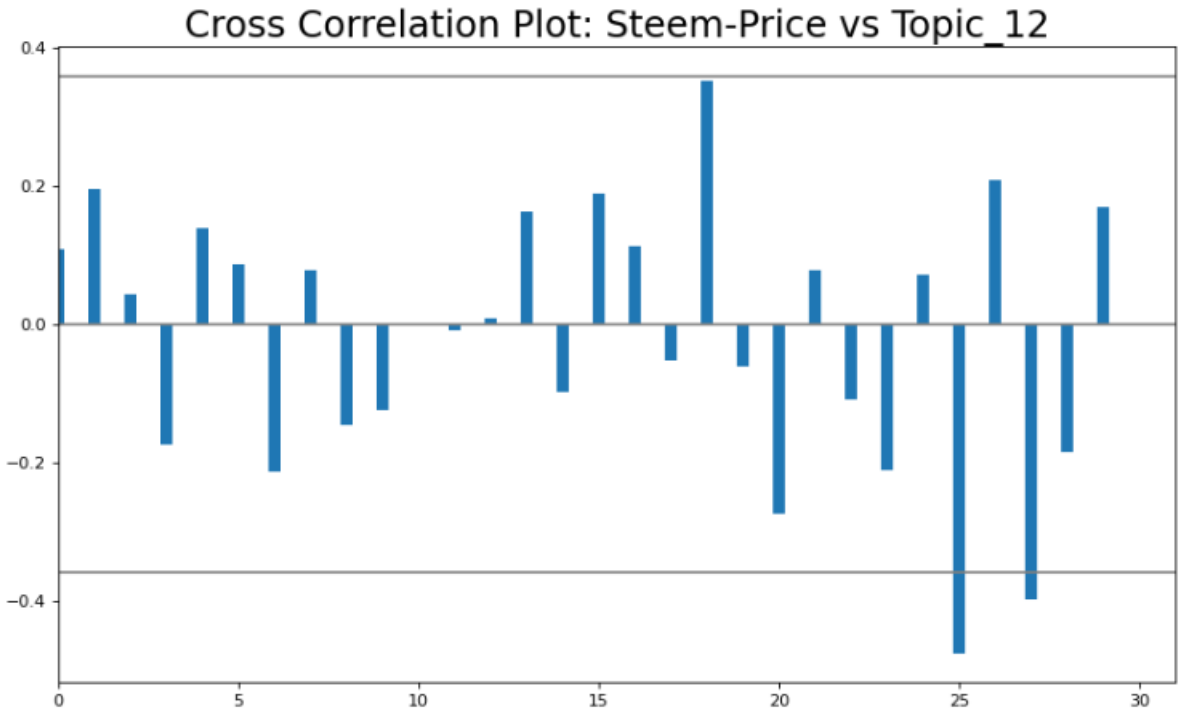

Figure 11: Python Output: Cross Correlation between Bitcoin related topics vs Steem Price growth

To understand better the relationship, we plot also the cross correlation. The highest similarity comes at lag 18 while the lowest at 25. The correlation analysis gives mixed results. Given this methodology, we cannot conclude that any topic is positively related to the price. However, given the keywords, we choose topics 8,9,12,13 to calculate the daily polarities.

Chosen topics and any posts having such topics are given to the tweet trained model. For topic and day, the polarity scores are averaged to get a daily value.

timestamp	8_upvote_post_steemit_comment	9_steemit_love_friend_year	12_bitcoin_full_story_source_br	13_market_money_company_report	close
2017-12-01	0.777747	0.887534	0.690516	0.550015	1.08
2017-12-02	0.870495	0.881517	0.759627	0.565658	1.20
2017-12-03	0.847345	0.835866	0.732898	0.558167	1.22
2017-12-04	0.849333	0.855866	0.701612	0.557061	1.43
2017-12-05	0.845052	0.863808	0.708850	0.559348	1.46
2017-12-06	0.853248	0.863204	0.683203	0.565437	1.28
2017-12-07	0.815539	0.856798	0.660821	0.568091	2.05
2017-12-08	0.847454	0.849944	0.659044	0.487638	1.94
2017-12-09	0.864842	0.866758	0.671777	0.561277	1.67

Figure 12: Python Output: Daily polarity result

The resulting matrix is provided in figure 12. We fed an LSTM model with two data matrices. First, with the one provided in figure 12 and the other with only the close price.

```
Text(0.5, 1.0, 'Steem Price Prediction')
```

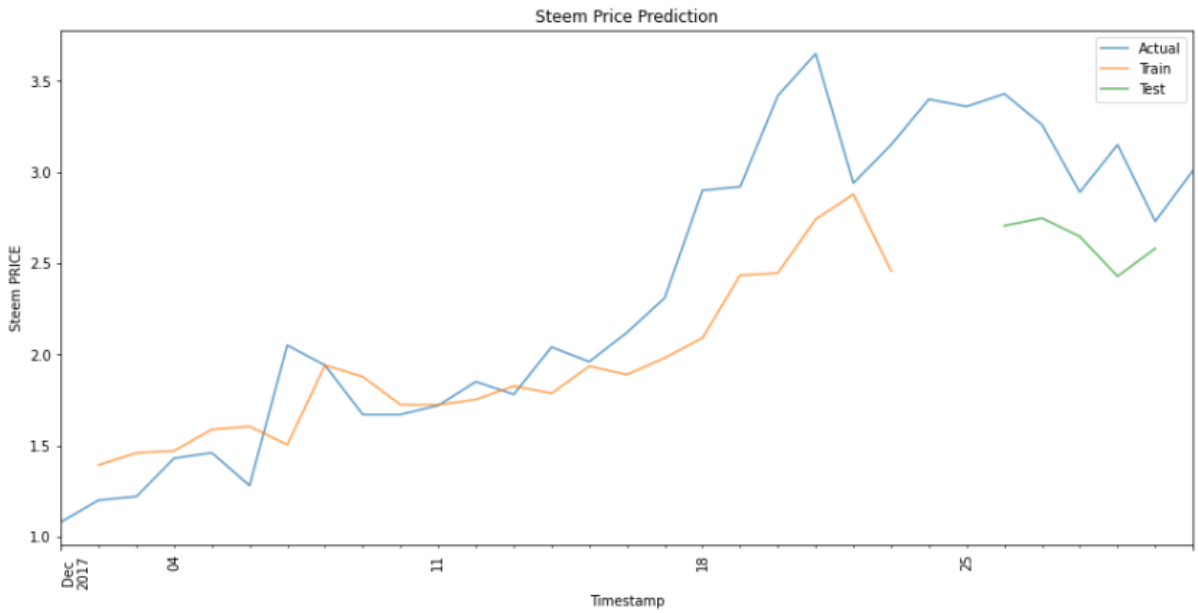

Figure 13: Python Output: LSTM prediction plot with only close value

Since we are using only 31 days of data and since generally deep learning models need huge amount of data, the result is not good

however our aim here is only to compare the model with and without textual information.

Figure 14: Python Output: LSTM prediction plot with only close value

In Figure 14, we present LSTM multivariate model. Here we fed the LSTM with only bitcoin related topics' polarity along with financial market's polarity. Topic 8 and 9 reduced the test accuracy.

	LSTM_close	LSTM_multivariate
testScore_RMSE	0.5269703880051597	0.3181908681057473
testScore_MAE	0.4699025077819825	0.30552508471455464
trainScore_RMSE	0.42362595946739007	0.31936864363159273
trainScore_MAE	0.30237215497277	0.24044375061630435

Figure 15: Python Output: LSTM prediction metric table

Figure 15 presents the LSTM result. We used Mean Absolute Error and Root Mean Square Error. Multivariate LSTM with textual information out-performed the base model in all the metrics over both test and training accuracy in this analysis.

5 Conclusion

We did a topic modelling on steemit texual posts. Among 14 topics, we considered only 4 of them for the polarity calculation. We looked at Pearson correlation and cross correlation tests but it did not provide us a useful result. Based on our intuition and some weekly significant correlations, we chose topic 8,9,12,13 for polarity calculation. To calculate the polarities, we trained a model on 1 millions of tweets. Tweets had labels and the dataset was balanced, therefore the polarity results were reliable. After calculating the polarities on choosen topics, we averaged them by topic and day. The resulting matrix is fed into an LSTM model. Multivariate LSTM with textual information outperformed the base model where we used only the close price values.