FINAL PROJECT - Cryptocurrency Analysis Engine

Xueshan Bai
Xinnan Chen
Kwang Bin Lee

PLEASE SEE READ ME ON HOW TO RUN OUR CODE

Github: https://github.com/klee166/crypto_comp_project

## JUSTIFICATION

When we as a group of two people were discussing about our project topic, our implementation plan seemed very overwhelming and looked like it requires a lot of works since we want to apply various techniques we learnt in and outside class to this projects (vector model, cosine similarity, Naive Bayesian classification, clustering and sentiments analysis). And, we decided to implement our program in two different languages such as Python and Perl. So, we were looking for a person who is really proficient in python language, and found a person who is good at it. So, we decided to form a group of three people.

## PROJECT DESCRIPTION

Today, cryptocurrencies have turned into a global phenomenon and are becoming more and more accessible. Cryptocurrency is not issued by a certain authority, and thus avoids the issue of the involvement of governments in currency. Because of this independence from outside influence, cryptocurrencies attract a lot of businesses and individuals, but sometimes their lack of knowledge on the subject can seriously impede them. Therefore, our final project takes on the challenge of analyzing and evaluating a variety of cryptocurrencies for people who are interested in learning more. We have implemented two versions of the application, using Perl and Python, respectively.

## DATA COLLECTION

We start our data collection on cryptocurrencies from a prestigious cryptocurrency evaluation site, CoinMarketCap and other sources like bitcoin-wiki. We have written a web crawler, web_robot.py, to browse through the site and extract useful information. We use the Requests, a Python HTTP library, to make HTTP requests. CoinMarketCap has links to each cryptocurrency listed on its page, which we traverse with our web crawler. We excluded URLs with bad status code (e.g., 404). For each good url, we look for contents with the “p” tag, because often useful information is contained in the paragraph. We then write the content to an output file called myfile.txt. We use “.I CRYPTOCURRENCY1_NAME” to mark the start of each cryptocurrency in myfile.txt.

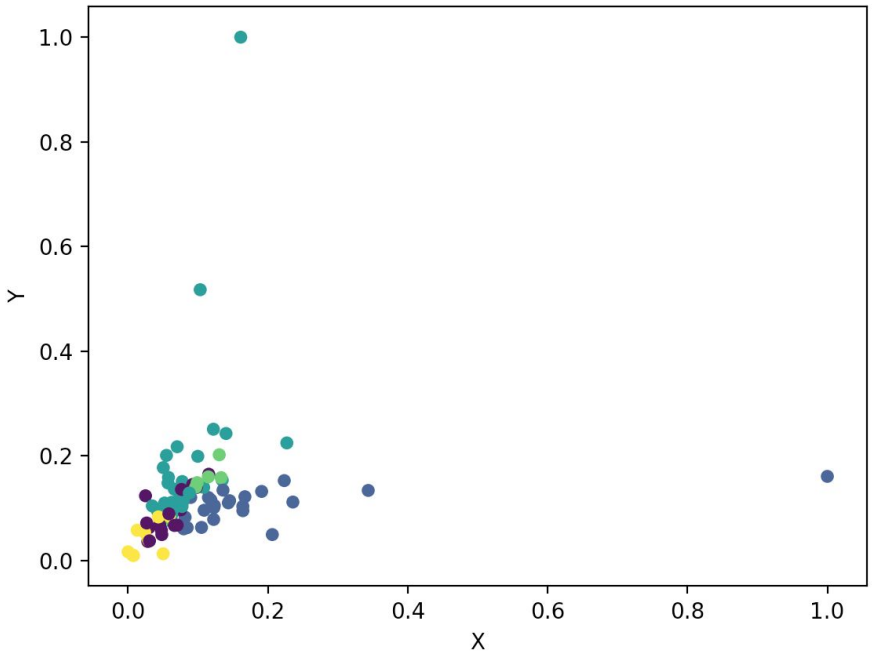
## DATA PREPROCESSING

Since the data in myfile.txt is pretty raw, we need to do some data preprocessing on it. For our python version, we use the NLTK package to stem and normalize the text. We remove the punctuation, convert to lower-case, and tokenize the words for a more understandable, easy-to-process format. For our perl version, we write stemmer.py and tokenizer.py to achieve similar goals and store the results in myfile.stemmed and myfile.tokenized. Then we use make_hist.prl to make these two files into histograms, which is saved in myfile.stemmed.hist and myfile.tokenized.hist. Additionally, we employ a list of stop words to make sure common words do not interfere with our computation of similarity.

**PYTHON CYPTOCURRENCY ANALYSIS APPLICATION**

The goal of our project is to provide the users a way to easily compare cryptocurrencies, identify cryptocurrencies that compete with each other(i.e. focus on similar topics) and find the best cryptocurrency among certain groups. We vectorize the cryptocurrencies and show how similar/dissimilar they are by computing the cosine similarity between all pairs of cryptocurrencies. This way, the users can clearly see the comparison of any vectors. Some users would also benefit from knowing the general opinion on a cryptocurrency. Therefore, we complete a sentiment analysis for each cryptocurrency and compute scores for each cryptocurrency vector in 2 dimensions, positive and negative, based on the terms contained in each vector.

We then performed k-mean clustering and hierarchical clustering to partition the vectors into k clusters. A HAC method is also implemented. We generate a plot (Fig. 1) for the users to visually see how the cryptocurrencies can be divided into different categories. Dots of the same color belong to the same category. They are considered to be competitors because they are very similar and attract the same group of investors. So, they are competing for investors. We can return to the users the best competitor in each category, which would be the cryptocurrency with the highest pos-neg (difference between the pos score and the neg score) value (See Figure 1-3).



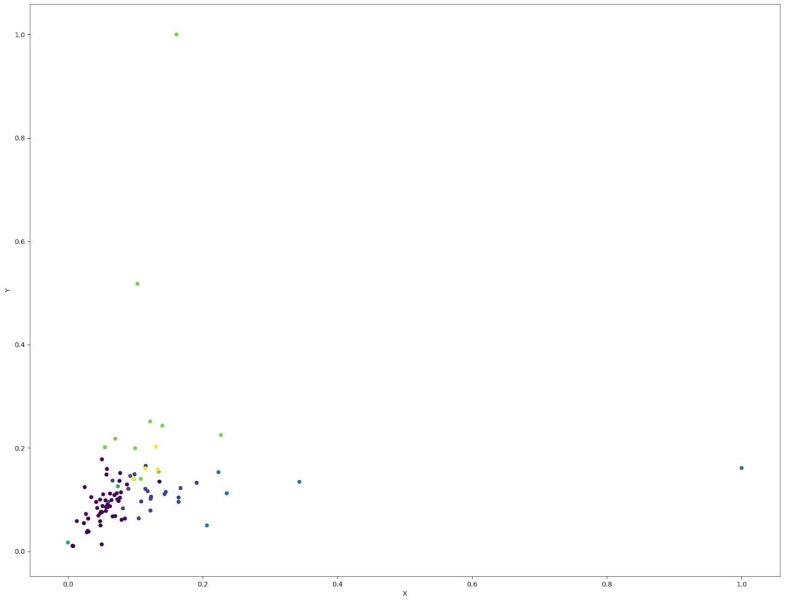



Figure 1

As our python program divides a group of currencies into separate clusters, it prints out each currency's name and which cluster it belongs (See Figure 1-2).

```
Mithril
3

Gas
3

MonaCoin
3

GXChain
1

Ethos
4

Skycoin
0

QASH
4

CyberMiles
1

Veritaseum
1
```

Figure 1-2

```
The Cluster No:
0
The Best Competitor:
Ethos

The Cluster No:
1
The Best Competitor:
Huobi

The Cluster No:
2
The Best Competitor:
Storm

The Cluster No:
3
The Best Competitor:
Litecoin

The Cluster No:
4
The Best Competitor:
Mithril
```

Figure 1-3

**PERL CYPTOCURRENCY ANALYSIS APPLICATION**

Since we have learned different ways to implement things in class, we have implemented another Perl version of our application in vector1.prl. In the Perl version, we give the users 2 options to compare cryptocurrencies.

```
============================================================
==     Welcome to the 600.466 Cryptocurrency Analysis Engine
==
== Total Documents: 91
============================================================

OPTIONS:
  1 = Find the similarity between two cryptocurrencies
  2 = Find cryptocurrencies most similar to a given one
  3 = Quit

============================================================

Enter Option:
```

A list of cryptocurrencies with index number is provided to the user. One option is to compare two specific cryptocurrencies. The users can choose two cryptocurrencies that they are interested in, and we compute the cosine similarity between the two vectors. We use a list of keywords that are especially meaningful to the field of cryptocurrency (p2p, scalable, transaction, etc.) More weights are given to these words, so we are really comparing two cryptocurrencies instead of just two documents on random topics. Fig.2 is a sample result of the comparison.

```
1st Document number: 2

2nd Document number: 20
============================================================
Vector Overlap          2         20    Product result
============================================================
maintain                1          1           1
assets                  1          1           1
machine                 4          2           8
part                    2          1           2
stored                  3          2           6
based                   4          1           4
mechanism               2          1           2
hold                    3          1           3
according               1          1           1
develop                 2          1           2
smart                  15          9         135
Contracts               1          1           1
standard                6          1           6
enables                 1          2           2
requires                1          1           1
built                   4          1           4
run                     7          2          14
```

*Figure 2*

Sometimes users are torn between two cryptocurrencies and want to see which one is better, which is what option 1 is for. Other times users might already have a cryptocurrency that they really like, and

they just want to see if there are any other cryptocurrencies like that. So, we provide another option to find k cryptocurrencies that are most similar to a cryptocurrency chosen by the user, where k is also given by the user. See Fig 3 for a sample run of this option.

```
Target Document/Query number: 2

Show how many matching documents (20): 5
Document to Document comparison

    ************************************************************
    Documents Most Similar To Document number 2
    ************************************************************
    Similarity    #      Cryptocurrency Name
    ==========   ====  ======================

     1.000000     2      Ethereum
     0.756831     17     Ethereum Classic
     0.450605     39     Augur
     0.433674     20     Qtum
     0.413310     23     OmiseGO
     0.401322     47     Status

Show the terms that overlap between the query and retrieved docs (y/n): y
============================================================
Vector Overlap                2          2       Docfreq
============================================================
hacking                       4          4       1
commencing                    1          1       1
Vitalik                       3          3       1
compete                       1          1       1
deploy                        1          1       1
utilize                       1          1       1
Hoskinson                     1          1       1
creates                       1          1       1
excluded                      1          1       1
```

## Bayesian Model

We also implemented a bayesian model in vector1.py, which classifies each cryptocurrency into one of the 5 groups discuss in this article(https://medium.com/swlh/a-better-taxonomy-for-cryptocurrencies-cbffd2e1b58c): Mode of Payment/Currency, Store of Value, Protocol Improvement, Coin-as-a-Service(CAAS), and Utility Token. (see picture below: each docnum represents one cryptocurrency and we print out the bayesian log likelihood for each category)

```
  Docnum    Mode of Payment  Store of Value  Protocol Improvement  Coin-as-a-Serv
ice   Utility Token
  ============================================================================
==================

1    -5615.47    -5309.63      -5595.90    -5633.41    -5965.41
2    -20359.79    -18773.26      -20157.52    -20059.73    -21327.77
3    -23462.07    -21801.92      -23238.07    -23139.98    -24603.00
4    -29651.43    -28261.14      -29406.64    -29686.19    -31595.05
5    -32116.63    -30701.10      -31825.81    -32043.07    -34191.47
6    -35019.29    -33825.22      -34696.00    -35122.39    -37481.73
7    -37278.74    -36032.79      -36932.63    -37288.19    -39859.57
8    -40313.85    -38980.00      -39872.70    -40247.20    -43031.12
9    -47819.62    -46345.85      -47315.47    -47639.63    -50890.64
10    -50758.42    -49211.71      -50212.88    -50506.22    -53080.33
11    -53483.11    -51795.51      -52870.97    -53122.46    -55889.46
12    -59704.64    -58693.89      -59048.05    -60173.72    -63387.81
13    -64078.26    -63800.71      -64114.23    -65164.76    -68851.99
14    -70663.58    -70312.26      -70670.63    -71508.57    -75784.84
15    -72402.14    -72045.51      -72353.40    -73218.97    -77585.52
16    -72921.04    -72579.90      -72863.38    -73719.39    -78110.87
17    -80957.81    -80498.58      -80795.95    -81689.82    -86556.88
18    -83738.51    -83352.82      -83558.05    -84593.32    -89598.02
```

## OVERALL EVALUATION

We evaluate our project with an empirical approach. We test the similarity for pairs of vectors that are known to be similar. For example, this article about cryptocurrency (https://medium.com/swlh/a-better-taxonomy-for-cryptocurrencies-cbffd2e1b58c) divides cryptocurrency into 5 categories: Mode of Payment/Currency, Store of Value, Protocol Improvement, Coin-as-a-Service(CAAS), and Utility Token. Litecoin and Bitcoin Cash are classified as Mode of Payment because they both have the crucial quality of being faster, more scalable, and fee-less transactions. We run our Perl application on this pair of cryptocurrencies, and the results are below in Figure 4(a). Bitcoin Cash has an index number of 4, and Litecoin has an index number of 6 in our application. As the results show, Bitcoin Cash indeed is the 2nd most similar to Litecoin out of the 91 cryptocurrencies that we have in our system.

```
Target Document/Query number: 6

Show how many matching documents (20): 5
Document to Document comparison

    ************************************************************
    Documents Most Similar To Document number 6
    ************************************************************
    Similarity    #      Cryptocurrency Name
    ==========   ====  ======================

     1.000000     6      Litecoin
     0.367216     26     Bitcoin Gold
     0.342885     4      Bitcoin Cash
     0.320608     1      Bitcoin
     0.296656     44     Bitcoin Private
     0.295853     31     Decred
```

*Figure 3*

In our Python version, the cosine similarity also shows the these 2 cryptocurrencies are very similar. They are also being clustered together using the k-mean clustering functions in our Python application.