Tawatchai (Andy) Holmes tho78 81662793

DATA301 Final Report

Summary

After the pandemic outbreak around the beginning of 2020, a spike of interest in the cryptocurrencies world started to blow up. People around the world started to be interested/invested in cryptocurrencies (BTC, ETH, etc.), causing the crypto market to blow up and push most crypto prices to reach all time high (BTC ~ $63,000, ETH ~ $4,000) a couple months after the outbreak in around march. This made me become interested in seeing how much the popularity of cryptocurrencies has risen after the covid outburst. In this Final Report, I will be analyzing the popularity/interest in the top 2 cryptocurrencies before and after covid outbreak, my hypothesis would be that the popularity of the top 2 cryptocurrencies increased after the covid outbreak as more people found out about cryptocurrencies after the outbreak. I will be using the Volume/Tone from GDELT Summary to compare the before and after, for comparison between before and after I will be using cosine similarity algorithm and will be creating graphs to visually see the difference (if any). I hope my results can prove/convince that there is a difference in the volume of interest before and after the outbreak alongside the volume of interest has risen as well.

Introduction

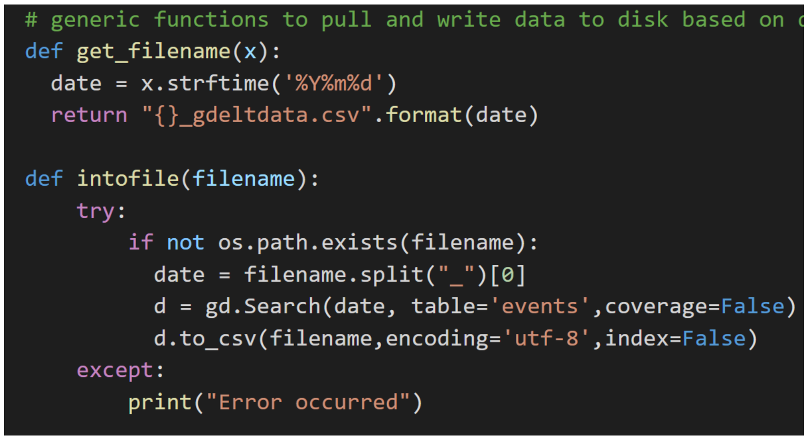
GDELT Summary is a powerful browser-based global multilingual and visual online news search platform. It is simple and easy to use, you can enter your search using simple keywords and dropdown filters, select the displays you want to see, etc., and click a single button to generate an instant new dashboard that summarizes global coverage of your topic that you are interested in. GDELT Summary allows you to choose different tones (Positive, Negative or Both) and provide the volume of mentions in the time frame that you choose, this is the data set that I will be getting from GDELT Summary using the URL of Volume and Tone. The algorithm that I went with is cosine similarity, cosine similarity is a measurement that shows the similarity between two vectors of an inner product space. It measures the direction between two vectors by cosine of the angle and determines whether the two vectors are pointing in roughly the same direction, It is also often used to measure document similarity in text analysis. I will be computing cosine similarity to the GDELT data that I have pulled (Volume and Tone).

As a person who has been interest/invested in cryptocurrency since 2017, I have went with the research question "***Analyzing the popularity/interest in the top 2 cryptocurrencies before and after covid outbreak***" as I have been affected largely during the covid outbreak with the massive drop in the crypto prices. I also became more interested in crypto after the covid outbreak as news sources everywhere were talking about the big crash, this made me become interested in how the covid outbreak affects the popularity of cryptocurrencies. I hope my research could help bring out some useful-information/confirmation about the popularity explosion of cryptocurrencies after the outbreak.

The research question that I went with is "***Analyzing the popularity/interest in the top 2 cryptocurrencies before and after covid outbreak***", my hypothesis would be that the popularity of BTC and ETH will increased after the covid outbreak as more people found out about cryptocurrencies after the outbreak. I will be comparing the Volume mentions and Tone of Bitcoin and Ethereum before and after the covid outbreak, I will be acquiring the data set that I need from GDELT Summary using the URL. Using the cosine similarity algorithm I can compute and compare the result of the before and after the outbreak of BTC and ETH.

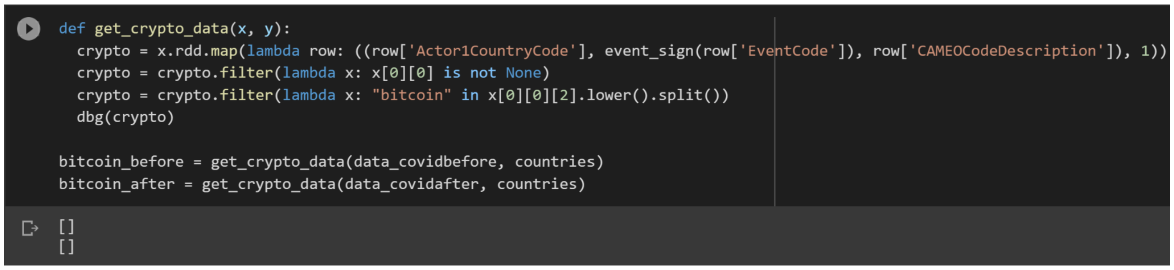
Experimental Design and Methods

I began experimenting with GDELT 2.0 to collect the dataset that I wanted to use. I used a function that was provided from the Sample code to pull data from specific dates (Figure 1), in this case I was pulling the all events data from the date 1st November 2019 - 31st January 2020 and 1st May 2020 - August 31 2020. I then made a list of countries that I would be interested in getting data from and those country were USA, NZL, CAN, GER and SGP, then I used the neutral_events, negative_events and positive_events code that was provided in the same Sample code for GDELT 2.0, I then wrote and run a function that would map and filter through the data set that I received. In Figure 2 we can see the filtered data result. It shows that it's an empty list meaning that there's no bitcoin keywords in the data set. I played around more with the filter in the Figure 2 function, but found no success, so I had to go back to GDELT Summary to pull data and scrape all the GDELT 2.0 code and redo this part.



```
# generic functions to pull and write data to disk based on d
def get_filename(x):
  date = x.strftime('%Y%m%d')
  return "{}_gdeltdata.csv".format(date)

def intofile(filename):
    try:
        if not os.path.exists(filename):
          date = filename.split("_")[0]
          d = gd.Search(date, table='events',coverage=False)
          d.to_csv(filename,encoding='utf-8',index=False)
    except:
        print("Error occurred")
```

Figure 1. GDELT 2.0 Search function.

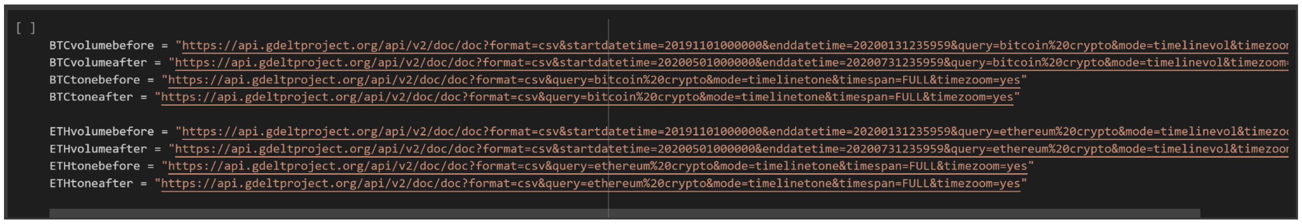


```
def get_crypto_data(x, y):
  crypto = x.rdd.map(lambda row: ((row['Actor1CountryCode'], event_sign(row['EventCode']), row['CAMEOCodeDescription']), 1))
  crypto = crypto.filter(lambda x: x[0][0] is not None)
  crypto = crypto.filter(lambda x: "bitcoin" in x[0][0][2].lower().split())
  dbg(crypto)

bitcoin_before = get_crypto_data(data_covidbefore, countries)
bitcoin_after = get_crypto_data(data_covidafter, countries)
```

```
[]
[]
```

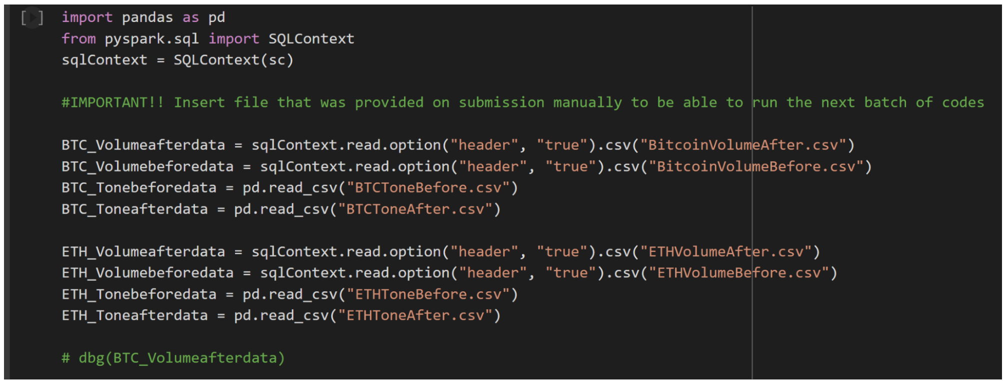
Figure 2. Code to map and filter through data.

I have decided to go with GDELT Summary to pull the dataset that I need, shown in Figure 3, as I can pull the datasets that I can get, such as the Volume of Mentions and Tones. I specify the URL to format as a csv file to be able to manually download the file using a print() statement for all the URLs, after I download the files I then upload the files onto Google Collab to be about to start doing map/filters. I read all of the files using Pyspark.sql and pandas, I had to used both as I was having trouble filtering and mapping the data later on in the code, so I have to read the Tone datasets for BTC and ETH as Panda Dataframe. I then created a function that will filter through the Tone datasets and select the dates that I want, shown in Figure 5, to filter through the dataset and pickout the dates similar to the Volume of Mention dataset and convert it back Pyspark.sql to be able to use rdd.map to get only just the value, as for the Volume of Mention I used rdd.map to select the appropriate columns that I need, shown in Figure 6. Lastly, I used a function that was in Lab 5 called changeint() to change the value from a string to a float for the Volume and Tone for both ETH and BTC, then used Lab 5 cosine similarity function to compute the before and after and print the result I got for both ETH and BTC, shown in Figure 7, and created a bar chart that will be display in the result section.



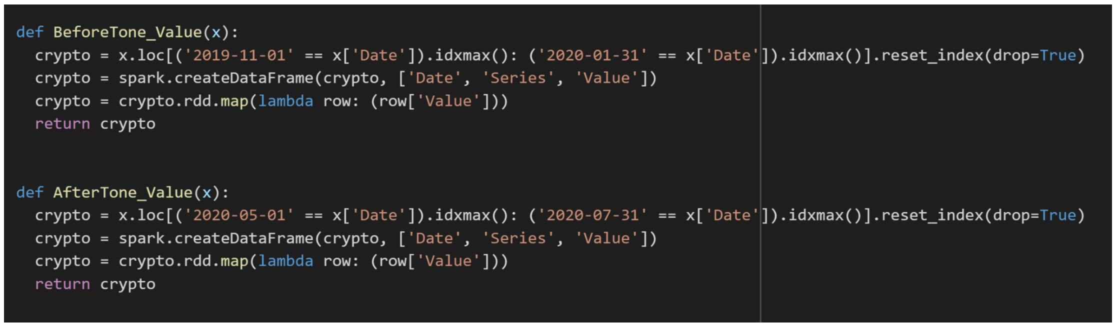
```
[ ]
    BTCvolumebefore = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&startdatetime=20191101000000&enddatetime=20200131235959&query=bitcoin%20crypto&mode=timelinevol&timezoor
    BTCvolumeafter = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&startdatetime=20200501000000&enddatetime=20200731235959&query=bitcoin%20crypto&mode=timelinevol&timezoom=
    BTCtonebefore = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&query=bitcoin%20crypto&mode=timelinetone&timespan=FULL&timezoom=yes"
    BTCtoneafter = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&query=bitcoin%20crypto&mode=timelinetone&timespan=FULL&timezoom=yes"

    ETHvolumebefore = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&startdatetime=20191101000000&enddatetime=20200131235959&query=ethereum%20crypto&mode=timelinevol&timezo
    ETHvolumeafter = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&startdatetime=20200501000000&enddatetime=20200731235959&query=ethereum%20crypto&mode=timelinevol&timezoor
    ETHtonebefore = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&query=ethereum%20crypto&mode=timelinetone&timespan=FULL&timezoom=yes"
    ETHtoneafter = "https://api.gdeltproject.org/api/v2/doc/doc?format=csv&query=ethereum%20crypto&mode=timelinetone&timespan=FULL&timezoom=yes"
```

Figure 3. GDELT Summary URLs.



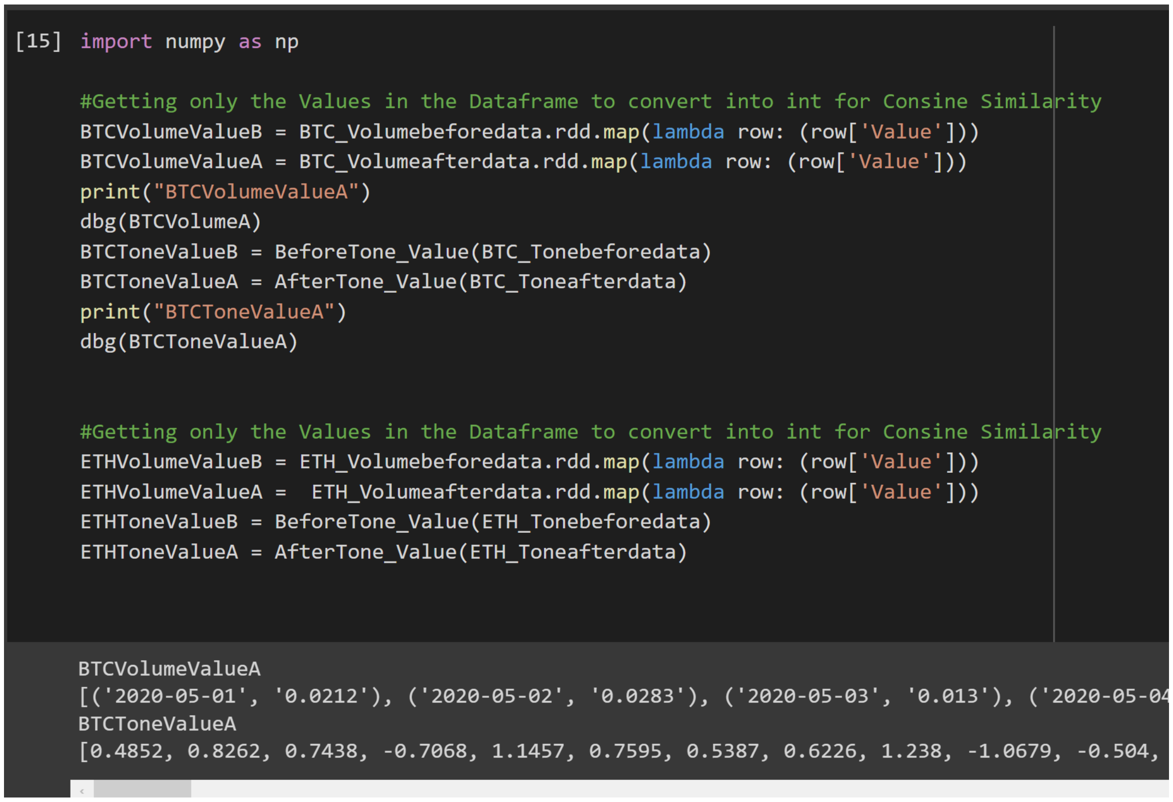
```
[ ] import pandas as pd
    from pyspark.sql import SQLContext
    sqlContext = SQLContext(sc)

    #IMPORTANT!! Insert file that was provided on submission manually to be able to run the next batch of codes

    BTC_Volumeafterdata = sqlContext.read.option("header", "true").csv("BitcoinVolumeAfter.csv")
    BTC_Volumebeforedata = sqlContext.read.option("header", "true").csv("BitcoinVolumeBefore.csv")
    BTC_Tonebeforedata = pd.read_csv("BTCToneBefore.csv")
    BTC_Toneafterdata = pd.read_csv("BTCToneAfter.csv")

    ETH_Volumeafterdata = sqlContext.read.option("header", "true").csv("ETHVolumeAfter.csv")
    ETH_Volumebeforedata = sqlContext.read.option("header", "true").csv("ETHVolumeBefore.csv")
    ETH_Tonebeforedata = pd.read_csv("ETHToneBefore.csv")
    ETH_Toneafterdata = pd.read_csv("ETHToneAfter.csv")

    # dbg(BTC_Volumeafterdata)
```

Figure 4. Reading files as sqlContext and Pandas.



```
def BeforeTone_Value(x):
  crypto = x.loc[('2019-11-01' == x['Date']).idxmax(): ('2020-01-31' == x['Date']).idxmax()].reset_index(drop=True)
  crypto = spark.createDataFrame(crypto, ['Date', 'Series', 'Value'])
  crypto = crypto.rdd.map(lambda row: (row['Value']))
  return crypto


def AfterTone_Value(x):
  crypto = x.loc[('2020-05-01' == x['Date']).idxmax(): ('2020-07-31' == x['Date']).idxmax()].reset_index(drop=True)
  crypto = spark.createDataFrame(crypto, ['Date', 'Series', 'Value'])
  crypto = crypto.rdd.map(lambda row: (row['Value']))
  return crypto
```

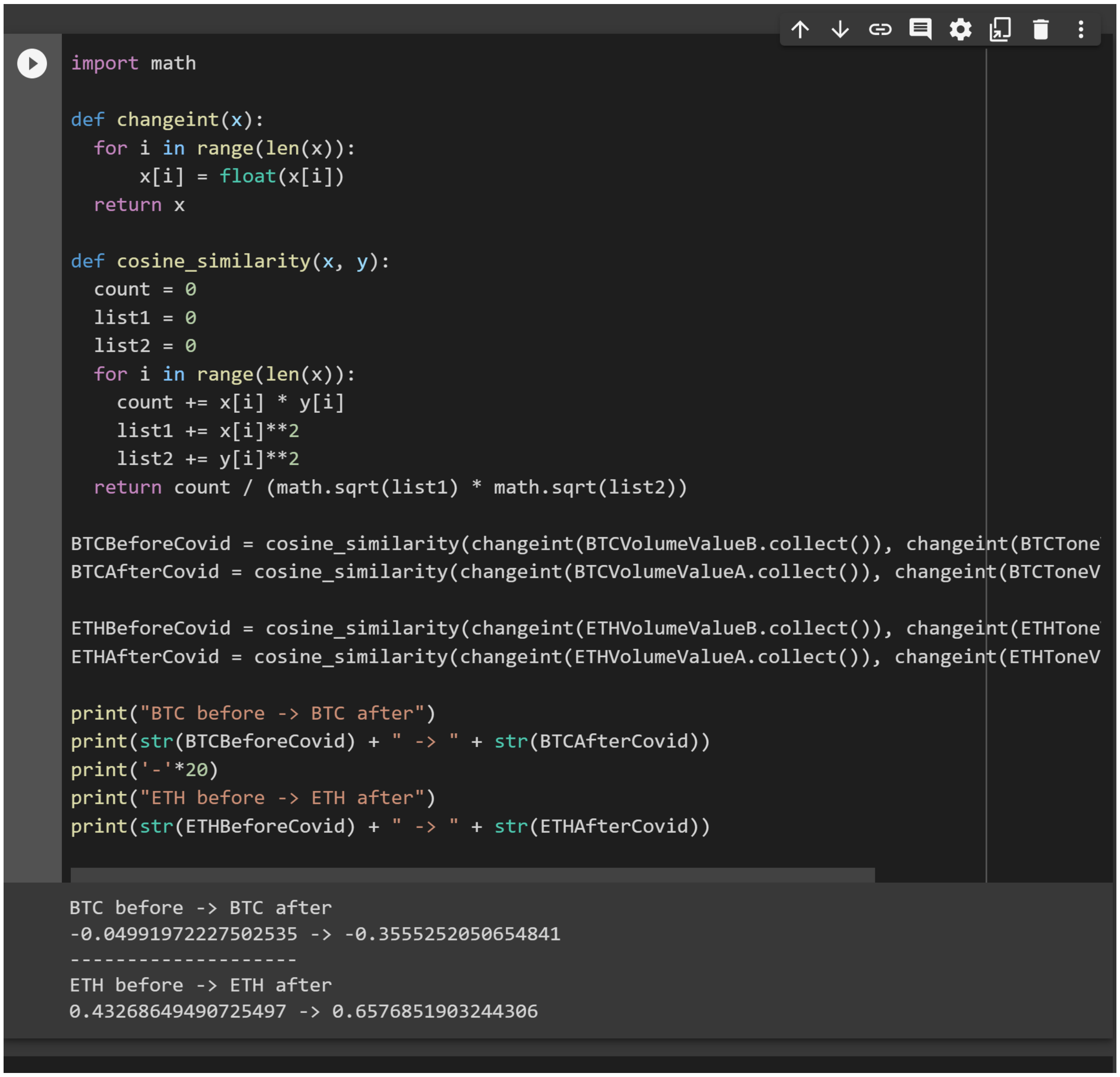
Figure 5. Functions to filter the Tone datasets.



```
[15] import numpy as np

     #Getting only the Values in the Dataframe to convert into int for Consine Similarity
     BTCVolumeValueB = BTC_Volumebeforedata.rdd.map(lambda row: (row['Value']))
     BTCVolumeValueA = BTC_Volumeafterdata.rdd.map(lambda row: (row['Value']))
     print("BTCVolumeValueA")
     dbg(BTCVolumeA)
     BTCToneValueB = BeforeTone_Value(BTC_Tonebeforedata)
     BTCToneValueA = AfterTone_Value(BTC_Toneafterdata)
     print("BTCToneValueA")
     dbg(BTCToneValueA)


     #Getting only the Values in the Dataframe to convert into int for Consine Similarity
     ETHVolumeValueB = ETH_Volumebeforedata.rdd.map(lambda row: (row['Value']))
     ETHVolumeValueA =  ETH_Volumeafterdata.rdd.map(lambda row: (row['Value']))
     ETHToneValueB = BeforeTone_Value(ETH_Tonebeforedata)
     ETHToneValueA = AfterTone_Value(ETH_Toneafterdata)
```

```
     BTCVolumeValueA
     [('2020-05-01', '0.0212'), ('2020-05-02', '0.0283'), ('2020-05-03', '0.013'), ('2020-05-04
     BTCToneValueA
     [0.4852, 0.8262, 0.7438, -0.7068, 1.1457, 0.7595, 0.5387, 0.6226, 1.238, -1.0679, -0.504,
```

Figure 6. Snippet of how function in Figure 5 and rdd.map was used with data shown.



```
import math

def changeint(x):
  for i in range(len(x)):
      x[i] = float(x[i])
  return x

def cosine_similarity(x, y):
  count = 0
  list1 = 0
  list2 = 0
  for i in range(len(x)):
    count += x[i] * y[i]
    list1 += x[i]**2
    list2 += y[i]**2
  return count / (math.sqrt(list1) * math.sqrt(list2))

BTCBeforeCovid = cosine_similarity(changeint(BTCVolumeValueB.collect()), changeint(BTCTone
BTCAfterCovid = cosine_similarity(changeint(BTCVolumeValueA.collect()), changeint(BTCToneV

ETHBeforeCovid = cosine_similarity(changeint(ETHVolumeValueB.collect()), changeint(ETHTone
ETHAfterCovid = cosine_similarity(changeint(ETHVolumeValueA.collect()), changeint(ETHToneV

print("BTC before -> BTC after")
print(str(BTCBeforeCovid) + " -> " + str(BTCAfterCovid))
print('-'*20)
print("ETH before -> ETH after")
print(str(ETHBeforeCovid) + " -> " + str(ETHAfterCovid))
```

```
BTC before -> BTC after
-0.04991972227502535 -> -0.3555252050654841
--------------------
ETH before -> ETH after
0.43268649490725497 -> 0.6576851903244306
```

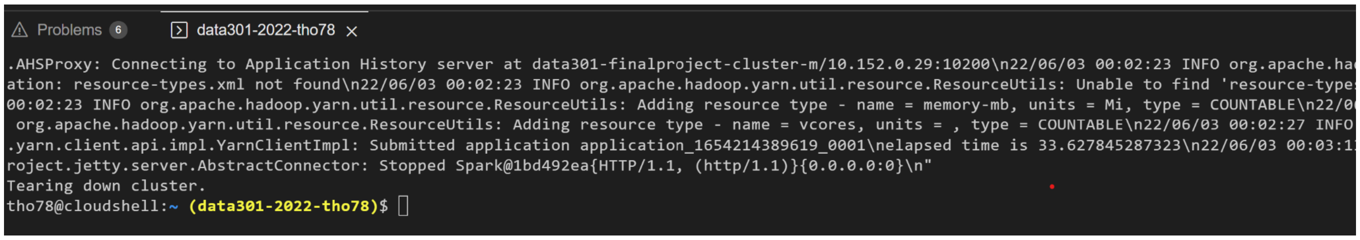
Figure 7. Cosine Similarity computing

- **Math library**: used to calculate cosine similarity in the cosine_similarity() function
- **Pyspark.sql library**: import SparkSession to convert Pandas dataframe back to pyspark dataframe
- **Pandas library**: used to filter through the dates
- **Def BeforeTone_Summary()**: function filter through dates with pandas dataframe and convert back to pyspark dataframe and used rdd.map to get only Date and the Value
- **Def AfterTone_Summary()**: function filter through dates with pandas dataframe and convert back to pyspark dataframe and used rdd.map to get only Date and the Value
- **Def BeforeTone_Value()**: function filter through dates with pandas dataframe and convert back to pyspark dataframe and used rdd.map to get only the Value

- **Def AfterTone_Value()**: function filter through dates with pandas dataframe and convert back to pyspark dataframe and used rdd.map to get only the Value
- **Def changeint()**: function to convert string values into float values
- **Matplotlib.pyplot library**: used to play bar chart and scatter plots to display the results
- **Def Filter_value()**: function to filter through the dataset and return all the needed column to be used for Google cloud

Result

I ran my code with different numbers of cores in google cloud (2, 4, 8, 16 vCPUs), shown in Figure 8, and time how long it took and plotted a graph to display the result.



```
.AHSProxy: Connecting to Application History server at data301-finalproject-cluster-m/10.152.0.29:10200\n22/06/03 00:02:23 INFO org.apache.ha
ation: resource-types.xml not found\n22/06/03 00:02:23 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Unable to find 'resource-type
00:02:23 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type = COUNTABLE\n22/0
 org.apache.hadoop.yarn.util.resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNTABLE\n22/06/03 00:02:27 INFO
.yarn.client.api.impl.YarnClientImpl: Submitted application application_1654214389619_0001\nelapsed time is 33.627845287323\n22/06/03 00:03:1
roject.jetty.server.AbstractConnector: Stopped Spark@1bd492ea{HTTP/1.1, (http/1.1)}{0.0.0.0:0}\n"
Tearing down cluster.
tho78@cloudshell:~ (data301-2022-tho78)$
```

Figure 8. Google cloud testing with 2 vCPU Cores result

I did a strong scalability test with a fixed problem size with my code and displayed the timing (Tseq/Tpar) to a line chart, shown in Figure 9. As seen in Figure 9, we can see that the Tseq/Tpar greatly increased when the core reached 8 and 16, this means that the code ran faster when the workers were increased, -20 seconds from 2 vCPUs timing to be exact, but there isn't much of a difference from 8 to 16 vCPUs.

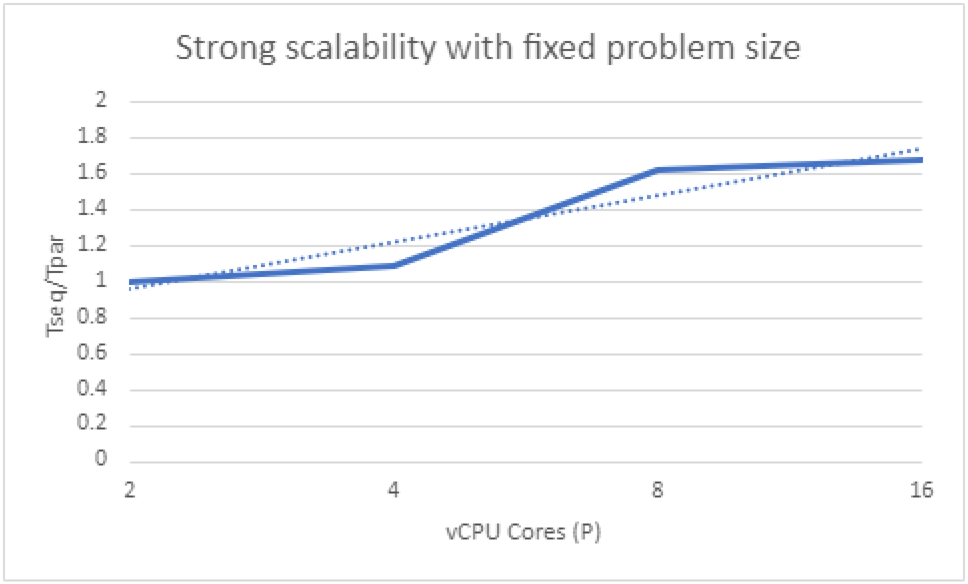


Figure 9. Strong scalability with a fixed problem size (3 months)

I also did a weak scalability test with increased problem size, shown in Figure 10. I increased the problem size from 3 months to 8 months instead and ran a test on google cloud. The result for 2 vCPU for weak scalability is doubled from the strong scalability, so this means that if the problem size increases massively it will take more time for the workers to compute the test. As seen in Figure 10, we can see that there isn't much difference for each of the vCPU cores, maybe just a slight increase in performance.

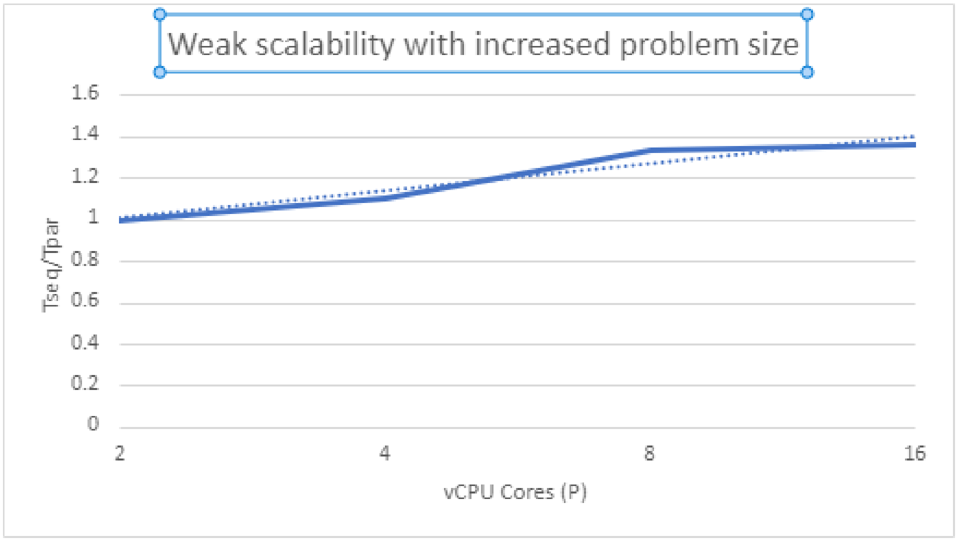


Figure 10. Weak scalability with an increased problem size (8 months)

In my hypothesis, I suggest that there will be an increase in the popularity of BTC and ETH after the covid outbreak. I have created a bar chart to display the result that I acquired from computing cosine similarity with the Before Volume and Tone to After Volume and Tone, the result that I got from one of the cryptocurrency is very interesting.

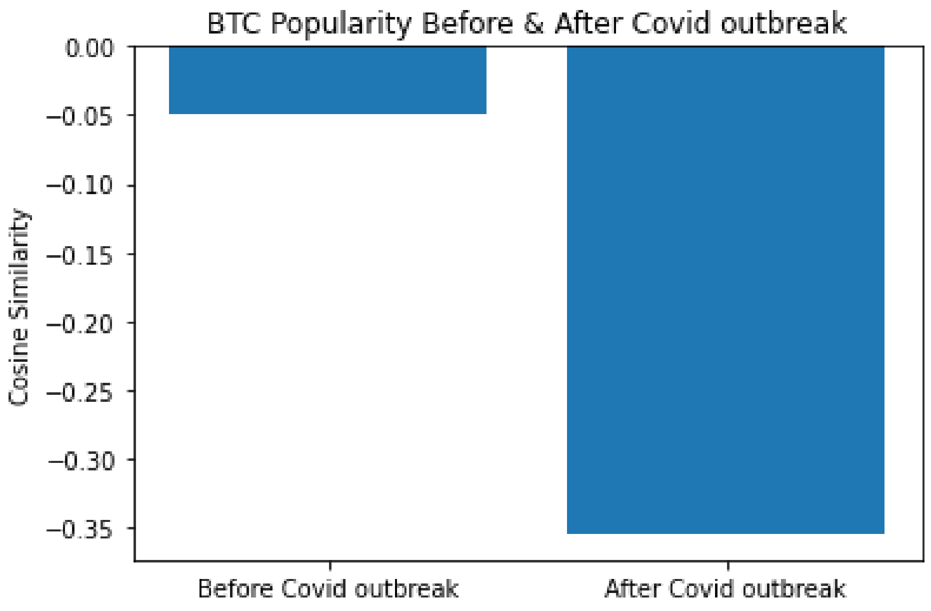


Figure 11. Barchart showing Cosine Similarity Before and After Covid for BTC

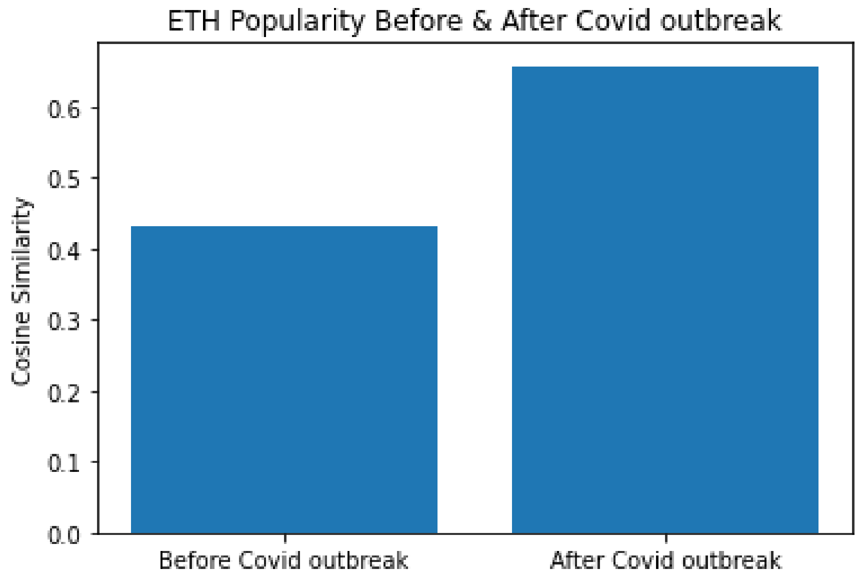


Figure 12. Barchart showing Cosine Similarity Before and After Covid for ETH

Looking at Figure 11, BTC cosine similarity before and after covid bar chart, we can see that the value of before covid outbreak is slightly below 0. This suggests that the Volume and Tone are independent, whereas, the value of after covid outbreak is heavily negative, -0.356 (2 d.p.) to be exact, this shows that the Volume and Tone are strongly opposite vectors, meaning either Volume of mentions is less than before covid or Tone is heavily negative after covid. Now we look at Figure 12, ETH cosine similarity before and after covid bar chart, we can see that before covid outbreak the Volume and Tone are quite similar, meaning both of the vectors are slightly similar (positive co-linear) vectors, but if we have a look at the value of after covid outbreak, it is heavily positive. This suggests that ETH had more similarity in Volumes of Mentions and Positive Tone after the covid outbreak compared to before covid outbreak. From the data that I received, it would suggest that my hypothesis is correct for ETH and could not come to a conclusion for BTC, because of the heavily negative cosine similarity. We can see in Figure 12 that after covid outbreak value is much higher than before covid outbreak, meaning the popularity of ETH has increased. And as for BTC there could be many factors that cause the value after covid outbreak to be heavily negative, but I cannot confirm nor deny that the popularity of BTC increased after covid outbreak.

Conclusion

I was able to answer my hypothesis/research question, but only for ETH, because I could not come to a conclusion for BTC as cosine similarity is a negative value. Figure 12 suggests that there's more Volume and Higher Tone for ETH after the covid outbreak, meaning that the cryptocurrency ETH has increased in popularity after the covid outbreak compared to before the covid outbreak. As for BTC, if we compare Figure 13 to Figure 14 we can see that after the covid outbreak there is a lot more Negative Tone, which would suggest why the cosine similarly after covid outbreak for BTC was heavily negative.

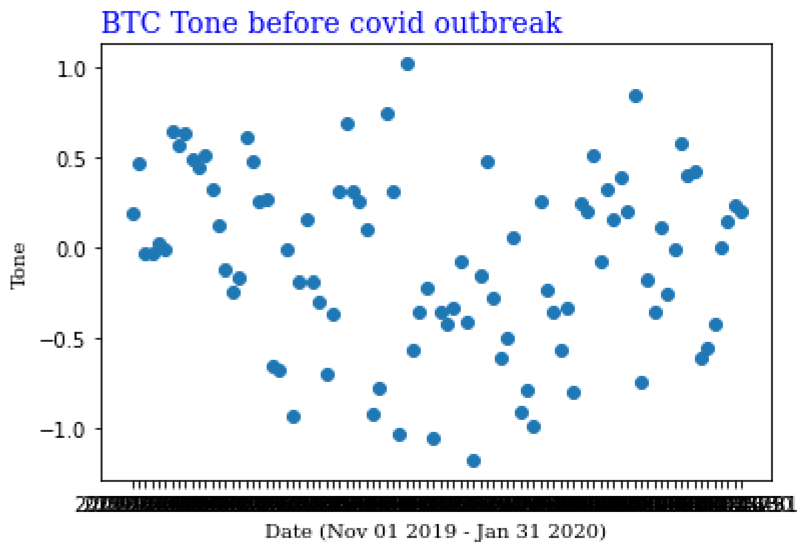


Figure 13. BTC tone plot before covid outbreak

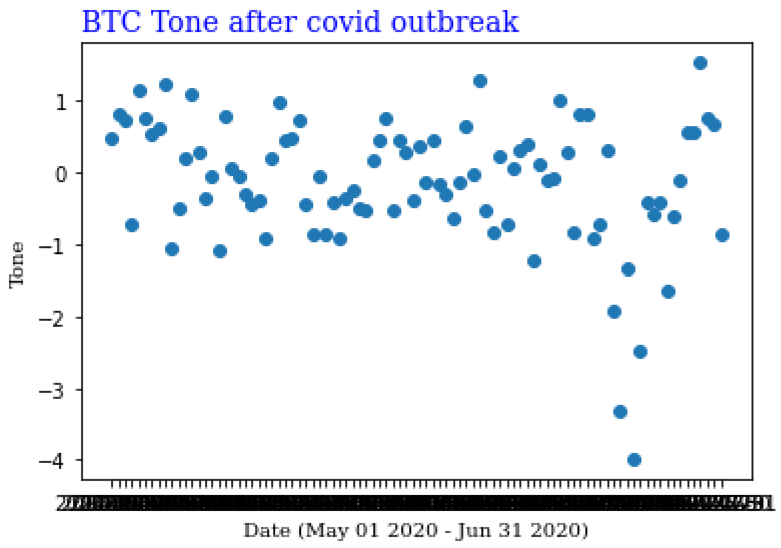


Figure 14. BTC tone plot after covid outbreak

My result suggests in the future if there were to be a major event, similar to covid, we can say that cryptocurrency would likely increase in popularity. This could benefit me to predict the cryptocurrency market/trend and make a great advantage of it, as I can predict that the cryptomarket will slowly grow/go back to normal after the major event has happened.

Future questions or direction that I would consider in the future of my project is to expand my knowledge/perspective of the cryptocurrency market/price trend, to look into more specific questions, such as, Price trend over a couple years/major events, Volume of trading before and after covid and Market Cap before and after covid. Lastly, add more algorithms from the course to improve performance of code and reliability/accuracy of my data.

Critique of Design and Project

Getting the datasets that I used for my project could be better, as I could not get specific datasets that I was wanting to use through GDELT Summary, such as price trend, volume of trading, market cap, etc. I only manage to get the Volume of Mentions and Tone from GDELT Summary, I feel in this case using GDELT Database 2.0 would be a better approach for automation/efficiency, if I managed to figure out how to use it properly, rather than manually downloading the file from the URL in GDELT Summary and uploading it onto the google collab.

Getting the datasets that I wanted from GDELT 2.0 didn't fully work for me, I tried different ways of approaching it and could not figure out how to work it, so I had to use GDELT Summary. I had tried using different ways to filter it to get the keywords and it seems to not work and I had a look at the csv file that was produced from the gd.search function for GDELT 2.0 and it only included major events around the world and not common events, so it didn't have anything that was related to cryptocurrencies.

Reflection

- GDELT Summary Explorer: Getting the URL
- Google Collab
- Cosine Similarity: Calculate the similarity of the two vectors
- Panda dataframe
- Matplotlib.pyplot: creating charts
- Google Cloud: computing number of cores against the project code
- Pyspark.sql
- Rdds

I learnt how to properly compute cosine similarity with real world datasets and utilize the result to answer my hypothesis, learnt how to properly extract datasets from GDELT Summary and how to properly get graphs from google cloud (with different number of cores) effectively. Lastly, Increase my knowledge of Pyspark.sql, Panda dataframes, and different algorithms that I could have used and converting Pandas dataframe to an rdd using SparkSession.builder.

Tawatchai (Andy) Holmes tho78 81662793

Reference:

List of student name that I have worked with:

- Ted Yang
- Zahid Khan

GDELT Summary:
https://blog.gdeltproject.org/announcing-gdelt-summary/

Cosine Similarity:
https://www.sciencedirect.com/topics/computer-science/cosine-similarity#:~:text=Cosine%20similarity%20measures%20the%20similarity,document%20similarity%20in%20text%20analysis.

Pyspark.sql documentation:
https://spark.apache.org/docs/2.4.0/api/python/pyspark.sql.html

Matplotlib.pyplot documentation:
https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.html

Pandas dataframe documentation:
https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

GDELT 2.0 Sample code I tried using to pull data (Not Successful):
https://colab.research.google.com/drive/1sTsl_-f2ipgzqM6htsVdKjf4MZ3Ds2CW#scrollTo=fVf4e7R5gM-n

GDELT Summary Sample code that i used to pull data successfully:
https://colab.research.google.com/drive/1hXAeG6yheFUQiHfc9Z5ISfNBqAQw47Dq

Cosine similarity/change int code that i used from Lab 5:
https://colab.research.google.com/drive/1SrwiPDWOMTUm2qGaQJW-AOacw7hJwoVH