Average volume (USD) trade by cryptocurrency

Introduction

The popularity of cryptocurrency has exploded over the last half of a decade due to it being a way to make digital transactions in a relevantly safe and secure way as well as being a worthwhile investment for early adopters of some of these specific digital currencies. Another reason for the rise in popularity of these virtual currencies is that the majority of cryptos are held on networks that are decentralised with the support of blockchain technology and the networks being distributed across multiple computers, this brings forth transactions being anonymous even for the parties involved and this decentralisation allows the process to exist outside the control of authorities such as banks or governments.

For the majority of people that take part in the trade of cryptocurrencies, they are not worried about the practical details and the specific technologies that each currency offers and uses but rather the profitability and potential that each currency holds. In this paper I will look at one of the key indicators of a cryptocurrency becoming widely adopted and profitable, this variable being the pure amount of volume of the currency traded.

Problem description

By looking at the volume of cryptocurrency traded we can infer if this particular currency will become profitable or not. We see that the greater volume of currency being traded shows that it is widely adopted and trusted by many users and very often means that the exchange prices are just and removes the possibility that the value of the currency being artificially manipulated for malicious purposes. **Hence, the goal of this project is to figure out which currencies are the most stable, widely adopted and therefore the most profitable to invest in by calculating their respective total average volumes traded.**

Due to the high volume and velocity of trades of these currencies occurring on a daily basis, possibly ranking in the millions, it makes sense to analyse this problem with the requirement of high performance computing methods such as using computer clusters with the processes being distributed between nodes. Such techniques will be used on my dataset to calculate the mean volume of each type of cryptocurrency which needs to be constantly updated day by day.

Associated dataset (EDA)

The data was produced by a user from scraping data from the CoinMarketCap Website and was uploaded to the kaggle website to the link: https://www.kaggle.com/datasets/philmohun/cryptocurrency-financial-data.
Below I will present a data dictionary with the variable name and a short description associated with it.

Variable	**Description**
Currency	Name of currency
Date	Date of transaction
Open	Price at the beginning of the day (USD)
High	Highest price of the day (USD)
Low	Lowest price of the day (USD)
Close	Price at the end of the day (USD)
Volume	Value of currency traded in 24 hour period (USD)
Market Cap	Number of coins in circulation multiplied by the price of the coin (USD)

The dataset contains 12 different cryptocurrency coins traded across the world and the frequency of transactions per coin will be provided in the table below

Currency	**Number of transactions**
binance-coin	2412
bitcoin	2412
bitcoin-cash	2412
bitcoin-sv	2412
cardano	2412
eos	2412
ethereum	2412
litecoin	2412

stellar	2412
tether	2412
tezos	2412
xrp	2412

We can see that the number of transactions scraped for this dataset were equally distributed between the 12 different crypto coins.

Below I will present a table with general summary statistics for the Volume of cryptocurrencies traded. It will present statistics such as the minimum, lower quartile, median, total mean of all coins, upper quartile and maximum values for the Volume variable.

Min.	**1st Qu.**	**Median**	**Mean**	**3rd Qu**	**Max**
0.000e+00	2.419e+05	5.213e+06	8.133e+08	1.555e+08	5.351e+10

Looking at these figures we can see that the range of volume traded varies drastically between 0 and roughly 53 billion dollars. With the average volume traded for all coins combined being roughly 810 million dollars.

Design

The high performance computation technique that will be used in this project to find the mean is called mapreduce. The mapreduce algorithm is a distributed processing technique that is built around the java programming language, it is used to process large amounts of data by distributing the data across various clusters. The nodes within these clusters process the data simultaneously using parallel computations allowing data that is high in volume and velocity to be processed in a relevantly short amount of time.

The mapreduce algorithm theory can be simplified in three main stages:

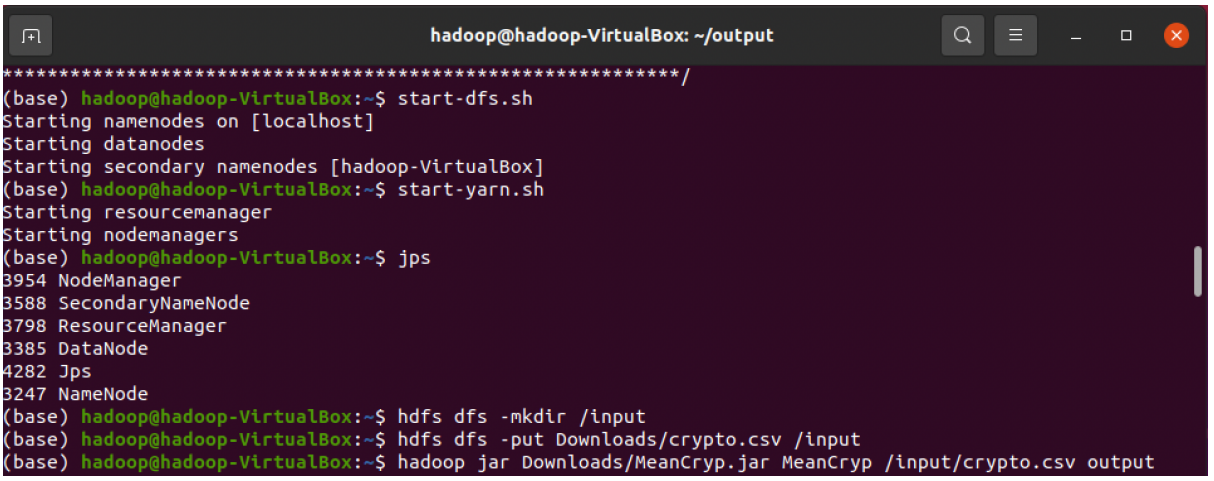
- Mapper stage: The input data is usually in a .txt or .csv file and is stored in either an input directory in the HDFS or stored locally in the operating system. The mapper's role is to process this input data by splitting and distributing the data to the various nodes. The mapper does this by viewing the input as a large group of key value pairs.

- Shuffling and sorting stage: This is the intermediate stage between the mapper and the reducer. This stage involves the movement of the key value pairs followed by the sorting process. This takes place by grouping the data from the mapper according to specific keys which are split and distributed to the reducer which are then sorted by these specific keys. Hence the reducer receives all values that correspond with the same key.

- Reducer stage: The reducer's role is to process and interpret the output data that came from the mapper. The specific output that the reducer receives are the key value pairs which have been grouped together according to the sorting and shuffling process mentioned above. This stage is also where the main function is defined and executed using the key value pairs, which in our case is calculating the average/mean volumes of trade by each cryptocurrency.

The prerequisites required before we can run the mapreduce procedure:

- The file we are using in our case is an excel file in the format of a .csv, which is named crypto.csv, this file is an edited version of the one found on kaggle as it only contains the two variables we are using (currency and volume) in our project
- Before we start the mapreduce procedure we need to start the Hadoop distributed file system daemons, i.e the datanodes and the namenodes
- As well as the YARN resource manager and the node managers
- Once DFS and YARN are both running, we make the HDFS directory required to execute the mapreduce procedure and copy the .csv data file into the distributed file system



The process of creating the mapper used in terms of the Hadoop distributed file system:

- In the mapper a DoubleWritable data type is created and named volume as well as a Text data type called currency
- To assign the correct variables to the newly created data types, the correct variables have to be read in from the .csv file
- A string variable is created and named ‘cur’ which is located in the 1st column of the .csv file, this string variable is then assigned to the Text data type we created earlier
- A double variable is created and named ‘vol’ which is located in the 2nd column of the .csv file, this double variable is then assigned to the DoubleWritable data type we created earlier
- For the output of the mapper, the key in this case is the currency Text data type and the value is the volume Doublewritable

The process of creating the reducer used in terms of the Hadoop distributed file system:

- For the input of the reducer, the key is still the currency Text data type but the value is now a list consisting of all the volumes Doublewritables
- An integer variable called size is created and is given a starting value of 0, also a double variable named SumofVolume is created and is also initialised with a starting value of 0
- A for loop is created which will count the volume values in the volume list and assign the count to the variable we created earlier called Size, as well as continuously accumulating each volume value added to the list and assign this accumulation value to the SumofVolume variable
- Then the reducer creates a new DoubleWritable variable called mean which divides the SumofVolume variable by the Size variable for each key value (i.e each currency)
- For the output of the reducer, the key is still the currency Text data type but the value is now the mean Doublewritable

Class type	**Input name**	**Input data type**	**Output name**	**Output data type**
Mapper	Key: Column position of Currency Value: Column position of Volume	Key: Longwritable Value: Text	Key: Currency type Value: Volume of trade	Key: Text Value: DoubleWritable
Reducer	Key: Currency type Value: List of Volumes	Key: Text Value: List of DoubleWritables	Key: Currency type Value: Mean of Volume	Key: Text Value: DoubleWritable

Implementation

In this section I will present the code that was used to create the three main classes to create the mapreduce process. The mapper, reducer and driver classes were created using hadoop and eclipse and were then compiled into a single jar file. After presenting the code, I will explain why this project required the use of high performance computing techniques such as mapreduce to produce a solution that effectively answers the research question and why normal techniques would not suffice.

Driver (Main class):

```
1  package org.myorg;
2
3  import org.apache.hadoop.conf.Configuration;..
12
13 public class MeanCryp
14 {
15  public static void main(String[] args) throws Exception
16  {
17      Configuration conf = new Configuration();
18      if (args.length != 3)
19      {
20          System.err.println("Usage: MeanDist <input path> <output path>");
21          System.exit(-1);
22          }
23      Job job;
24      job=Job.getInstance(conf, "Mean Cryp");
25
26      job.setJarByClass(MeanCryp.class);
27
28      //the input and output paths for the job
29      FileInputFormat.addInputPath(job, new Path(args[1]));
30      FileOutputFormat.setOutputPath(job, new Path(args[2]));
31
32     //the Mean volume mapper and Mean volume reducer for the job
33      job.setMapperClass(MeanCrypMapper.class);
34      job.setReducerClass(MeanCrypReducer.class);
35
36     //output variable types are Text and Doublewriteable respectively
37      job.setOutputKeyClass(Text.class);
38      job.setOutputValueClass(DoubleWritable.class);
39
40     //delete output folder if it already exists in the HDFS from previous projects
41      FileSystem hdfs = FileSystem.get(conf);
42      Path outputDir = new Path(args[2]);
43      if (hdfs.exists(outputDir))
44      hdfs.delete(outputDir, true);
45
46      //check the status of the job (completed or not) and exit when it is done
47      System.exit(job.waitForCompletion(true) ? 0 : 1);
48    }
49 }
50
```

Mapper:

```java
package org.myorg;
// Importing the libraries that contain the classes and methods needed in the Mapper class
import java.io.IOException;

public class MeanCrypMapper extends Mapper<LongWritable, Text, Text, DoubleWritable>

{
     //volume of trade is a large integer, it is converted to DoubleWriteable to be used in Hadoop
    //currency is a string of characters
     private static DoubleWritable volume = new DoubleWritable(0);
     private Text currency = new Text();
     @Override

     public void map(LongWritable key, Text value, Context context)
             throws IOException, InterruptedException
             {
         String[] line = value.toString().split(",");

        //currency name is located in the 1st column
         String cur = line[0];

         currency.set(cur);

        //volume is located in the 2nd column
         double vol = Double.parseDouble(line[1].trim());

         volume.set(vol);

         context.write(currency, volume);
        }
}
```

Reducer:

```java
package org.myorg;
// Importing the libraries that contain the classes and methods needed in the Reducer class
import java.io.IOException;

public class MeanCrypReducer extends Reducer<Text, DoubleWritable, Text, DoubleWritable>
{
    ArrayList<Double> volumeList = new ArrayList<Double>();

    @Override

    public void reduce(Text key, Iterable<DoubleWritable> values, Context context)
             throws IOException, InterruptedException
    {
        //create a integer variable called Size and start it with 0
                int Size = 0;

        //create a double variable called SumofVolume and start it with 0
        double SumofVolume=0.0;

        //The for loop will go through all the volume values passed to the reducer
        //from the mapper and do two things: 1) store the volume values in the
      //volumeList arraylist and accumulate the volume values and the associated group size values in the
    //variables SumofVolume and Size that we created earlier
        for (DoubleWritable value : values)
        {

        SumofVolume += value.get();
        Size += 1;

        }

        //calculate the average volume
        double mean = SumofVolume/Size;

        //transfer each commodity and the Mean Volume to the output file
        context.write(key, new DoubleWritable(mean));
     }
}
```

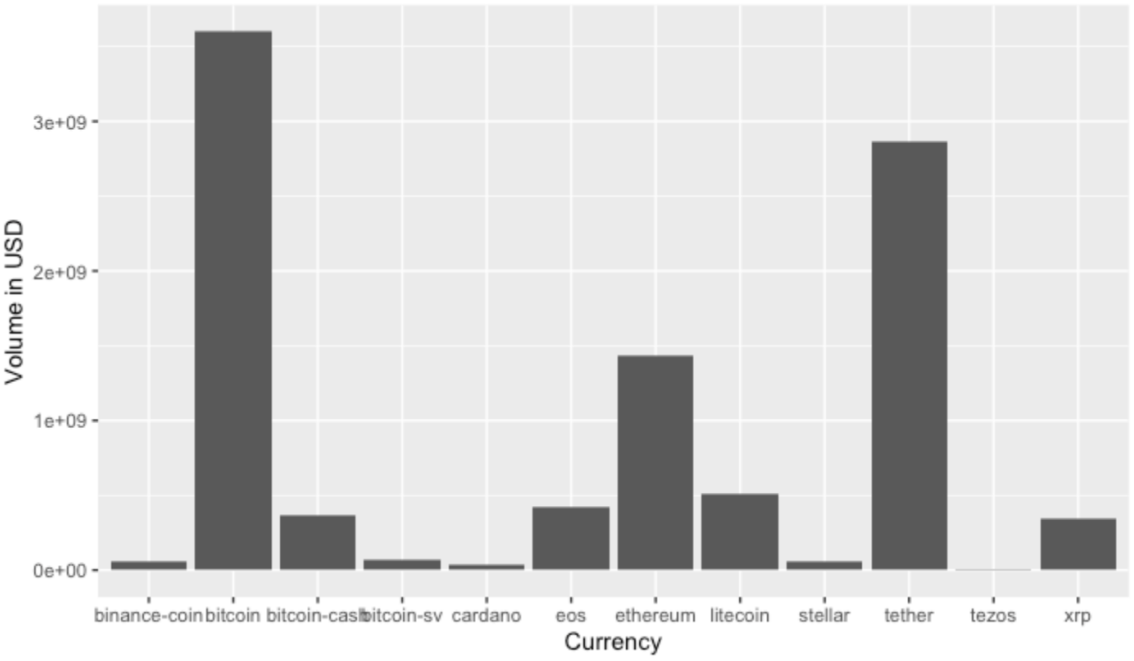
The main reason why this project was done using Hadoop clusters was because the nature of the research question requires us to process and evaluate large amounts of data that is constantly being updated by the day, hours, minutes and even seconds. This dataset looks at the history of the transactions of the 12 most popular cryptocurrencies and how they have been exchanged. This dataset is a very small batch of data with roughly 29,000 separate entries, which is only 2 hours worth of transactions just for bitcoin alone, as it is widely accepted that on average 400,000 transactions of bitcoin take place during a single day. So with 14,000 separate transactions happening over a single hour just for bitcoin it makes sense to use high performance computing techniques such as mapreduce on Hadoop clusters when processing the daily transaction data for the most popular cryptocurrencies in the world.

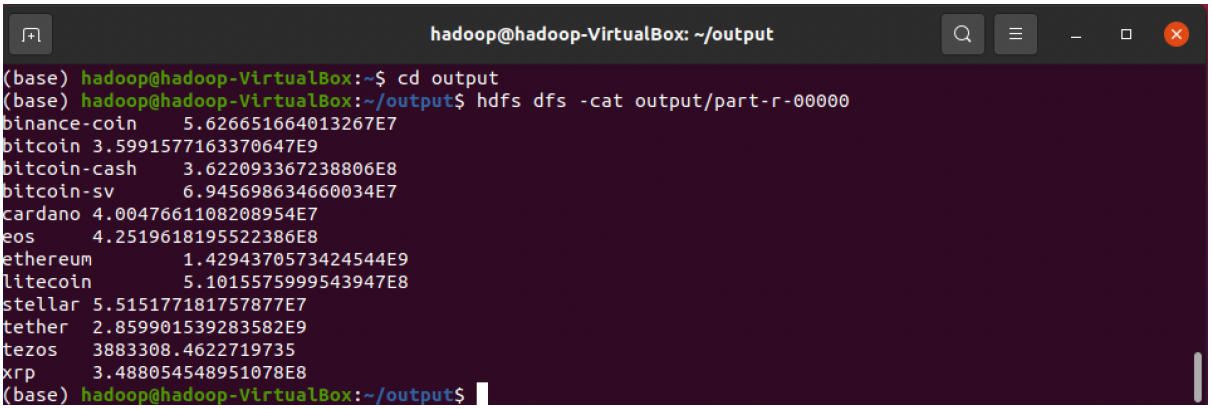
Hadoop clusters are essential for this task as discussed above, as there are millions of cryptocurrency transactions that take place on a daily basis. So programmes like R studio running on a single computer would have no chance to process and handle even a single batch worth 1 hours worth of this high volume and high velocity data. Hence clusters are the only plausible way to handle and manage this level of data.

The data processing would have to be split up between many clusters around the world as the amount of data is not just very fast and high in volume, it is also coming from various sources as cryptocurrencies are traded and stored in over 500 different exchange platforms and cryptocurrency wallets. So very large amounts of clusters would have to communicate and work in parallel with one another to keep up with the newest data as well as to manage and backup the archived data.

Results and evaluation

In this section I will discuss the results of running the mapreduce algorithm on the Hadoop cluster and present the average volume traded in USD (dollars) for each cryptocurrency and hence decide which of these cryptocurrencies are the most stable to invest in and hence most likely to be profitable due to their popularity and world wide adoption.





From the barchart it is quite obvious that there are 2 cryptocurrencies that have substantially higher average volumes than the rest. Bitcoin has the greatest amount of total average volume traded at a price of roughly 3.59 billion dollars, followed by tether with the total average traded at 2.85 billion dollars and then following a large gap the cryptocurrency with

the third highest amount of total average volume traded is ethereum with a price at 1.42 billion dollars.

Below, I will present the results of the average volume traded by each cryptocurrency in a table:

Currency	**Mean Volume traded (USD)**
binance-coin	5.626651664013267E7
bitcoin	3.5991577163370647E9
bitcoin-cash	3.622093367238806E8
bitcoin-sv	6.945698634660034E7
cardano	4.0047661108208954E7
eos	4.2519618195522386E8
ethereum	1.4294370573424544E9
litecoin	5.1015575999543947E8
stellar	5.515177181757877E7
tether	2.859901539283582E9
tezos	3883308.4622719735
xrp	3.488054548951078E8

The results show that people should invest in either bitcoin, tether and ethereum as they are the most stable cryptocurrencies to invest in and are the most widely adopted across the world according to the metric of the average volumes traded.