CIS-6930
HW2: Exploring Bitcoin transactions

Name: **Vejay Mitun Venkatachalam Jayagopal** **UF-ID:** **3106-5997**

I have worked with the tailored Data set with 212,576 blocks

Instruction run the file

Step 1: Install Anaconda console and navigator from https://www.anaconda.com/distribution/

Step 2: In the Anaconda console we need to install pandas by using "conda install pandas".

Step 3: Then go to the Anaconda Navigator installed in the start and set the environment path to the data's where they are locally stored(Compressed Data Set).

Step 4: Paste the given file along in the path to get access.

Step 5: Launch the Jupyter Notebook from the Anaconda Navigator and load the file.

Step 6: Run each line by pressing the play button on the top left pane or by pressing "Shift + Enter".

Part 1: Transactions analysis

Provide your answer to the following questions:

1. What is the number of transactions and addresses in the dataset?

Solution

Number of Transactions: **10000055**

Command:

```
file_path = 'C:/Users/prasi/OneDrive/Desktop/Vejay/Spring 2020/BlockChain/hw/HW2/DATA/compressed/data/bh.csv'
data = pd.read_csv(file_path, sep='\t', header=None)
data.columns = ['blockID','hash', 'block_timestamp','n_txs']
data['n_txs'].sum()
```

Number of Address: **8385065**

Command:

```
file_path_add = 'C:/Users/prasi/OneDrive/Desktop/Vejay/Spring 2020/BlockChain/hw/HW2/DATA/compressed/data/addresses.csv'
data_add = pd.read_csv(file_path_add, sep='\t', header=None)
data_add.columns = ['addrID','address']
data_add['address'].count()
```

2. What is the Bitcoin address that is holding the greatest amount of bitcoins? How much is that exactly? Note that the address here must be a valid Bitcoin address string. To answer this, you need to calculate the balance of each address. The balance here is the total amount of bitcoins in the UTXOs of an address.

   Solution

   Address with maximum number of bitcoins: 1933phfhK3ZgFQNLGSDXvqCn32k2buXY8a
   (Address ID :1083442)

   Maximum number of bitcoins : 11111100000000.0 **Satoshis**
   (OR)
   111111.00000000 bitcoins

   Command:

```
import pandas as pd
import numpy as np
filepath_out = 'C:/Users/prasi/OneDrive/Desktop/Vejay/Spring 2020/BlockChain/hw/HW2/DATA/compressed/data/txout.csv'
filepath_in = 'C:/Users/prasi/OneDrive/Desktop/Vejay/Spring 2020/BlockChain/hw/HW2/DATA/compressed/data/txin.csv'
data_in = pd.read_csv(filepath_in, sep='\t', header=None)
data_out = pd.read_csv(filepath_out, sep='\t', header=None)
data_in.columns = ['txID', 'input_seq', 'prev_txID', 'prev_output_seq', 'addrID', 'sum']
data_out.columns = ['txID', 'output_seq', 'addrID', 'sum']
sum_in = pd.DataFrame(data_in.groupby(['addrID'])['sum'].sum())
sum_out = pd.DataFrame(data_out.groupby(['addrID'])['sum'].sum())
result = pd.merge(sum_in, sum_out,how='right',on = 'addrID')
result=result.fillna(0)
result['balance'] = result['sum_y'] - result['sum_x']
result['balance'].max()                                    // gets max balance sathoshis

result.loc[result['balance'].idxmax()]           //gets the address ID
```

```
sum_x      0.000000e+00
sum_y      1.111110e+13
balance    1.111110e+13
Name: 1083442, dtype: float64
```

3. What is the average balance per address?

   Solution

   Average Balance per Address : 125990615.08928594

   Command:

```
Sum_balance = result['balance'].sum()
Avg_balancePerAddress = Sum_balance / total_no_address
```

4. What is the average number of input and output transactions per address? What is the average number of transactions per address (including both inputs and outputs)? An output transaction of an address is the transaction that is originated from that address. Likewise, an input transaction of an address is the transaction that sends bitcoins to that address.

Solution

Average number of inputs per address: **2** (2.6534393908097846**)**

Command:

```
file_path_in = 'C:/Users/prasi/OneDrive/Desktop/Vejay/Spring
2020/BlockChain/hw/HW2/DATA/compressed/data/txin.csv'
data_in = pd.read_csv(file_path_in, sep='\t', header=None)
data_in.columns = ['txID','input_seq','prev_txID','prev_output_seq','addrID', 'sum']
df_transactions_count = data_in['txID'].count()
ad_in = pd.DataFrame(data_in.drop_duplicates('addrID'))
addr_count = ad_in['addrID'].count()
avg_trans_perAddr = df_transactions_count / addr_count
```

Average number of outputs per address: **2** (2.77479281307554**)**

Command:

```
file_path_out = 'C:/Users/prasi/OneDrive/Desktop/Vejay/Spring 2020/BlockChain/hw/HW2/DATA/compressed/data/txout.csv'
data_out = pd.read_csv(file_path_out, sep='\t', header=None)
data_out.columns=['txID','output_seq','addrID','sum']
df = pd.DataFrame(data_out.drop_duplicates('addrID'))
out_addr_count= df['addrID'].count()
no_trans_out = data_out['txID'].count()
avg_no_out_transPerAddr = no_trans_out / out_addr_count
```

Average number of transactions per address: **5 (**5.180088850874192**)**

Command:

```
con_tab = pd.concat([data_out, data_in])
totAd = pd.DataFrame(con_tab.drop_duplicates('addrID'))
tot_no_of_address = totAd['addrID'].count()
Avg_transPerAddress =  (df_transactions_count + no_trans_out) / tot_no_of_address
```

5. What is the transaction that has the greatest number of inputs? How many inputs exactly? Show the hash of that transaction. If there are multiple transactions that have the same greatest number of inputs, show all of them.

Solution

Greatest number of inputs: 1312

Transaction ID: 7553001

Hash of the Transaction: 9621b3c67f9bddd3de65fafc488087b8f2b40b638e3a06209a904c66c0b32982

Command

```
data.loc[data['n_inputs'].idxmax()]

    txID          7553001
    blockID        201341
    n_inputs         1312
    n_outputs           3
    Name: 7553001, dtype: int64
```

6. What is the average transaction value? Transaction value is the sum of all outputs' value.

Solution

The Average Transaction Values: 12315588064.03543

Command:

```
df = pd.DataFrame(data_out.drop_duplicates('txID'))
no_trans_out = df['txID'].count()
out_sum= data_out['sum'].sum()
average_trVal=out_sum/no_trans_out
print(average_trVal)
```

7. How many coinbase transactions are there in the dataset?

Solution

Number of Coinbase transaction: 212576

Command:

```
data.loc[data['n_inputs'] == 0].count()

txID         212576
blockID      212576
n_inputs     212576
n_outputs    212576
```

```
dtype: int64
```

8. What is the average number of transactions per block?

Solution:

The average number of transactions per block: 47 (47.04225782778865)

Command:

```
total_transactions = data['n_txs'].sum()
number_of_blocks = data['blockID'].count()
Average_transaction_per_block = total_transactions / number_of_blocks
```

Part 2: Address de-anonymization

1. How many users are there in the dataset?

Solution:

Total number of Users per Data set :
Unique address (joint Control): 476884

2. Answer questions 2, 3, and 4 in part 1 by replacing "address" with "user". Note that each user is identified by the addresses that are owned by him/her. Thus, in answering question 2 (i.e., the user who is holding the greatest amount of bitcoins), you need to list all the user's addresses.

2.a) Listing all user's address holding the greatest number of bitcoins

Solution:

User Addresses holding largest number of bitcoins:

addrID	address
6851763	17SzMdJbw6wR8b4HzmFCn1kDhEbexbLPXK
6851857	1FxVdS6c1HSYDsMEHch8GkfegnuuPPrkoJ
6849838	1AY4dizXEGJA8mQKJFBcuEAResGt4uRZTV
6850395	1NnqM24fFeAGf7NWxmhhFkQAciPqeWo3L

Maximum no of bitcoins: **6010245309814 satoshis**

2 b) What is the average balance per users?

Solution:

Average balance per address: `250016268.66886467`

2 c) what is average number of input and output transaction per user ? Average number of transactions per address?

Solution:

Average number of input transactions per user: 4 (4.773078851851422)

Average number of output transactions per user: 5 (5.506309695866577)

Average number of transactions per user: 10 (10.279388547718)

3. Give the hash of the transaction sending the greatest number of bitcoins to the user who is holding the greatest balance.

Solution:

Hash of the transaction sending the greatest number of bitcoins:
70d46f768b73e50440e41977eb13ab25826137a8d34486958c7d55c5931c6081(TXID : 922967)