INFORMATION RETRIEVAL
Assignment 3

Group Members :
1. Abhishek Nigam (MT22003)
2. Dhananjay Bagul (MT22028)
3. Wilson Radadia (MT22091)

❖ **Question 1:**

- **Necessary Libraries:**

● **math**: provides standard mathematical constants and functions**.**
● **pandas:** It offers capabilities for cleaning, combining, and reshaping data as well as data structures for effectively storing and handling huge datasets.
● **np**: used for working with arrays. It also has functions for working in the domain of linear algebra, fourier transform, and matrices.
● **NLTK**: The Natural Language Toolkit (nltk) is used for performing tasks such as: Tokenization, Part-of-speech tagging, Sentiment Analysis,Stemming and Lemmatization etc.
● **numpy :** It offers functions for mathematical operations like linear algebra, Fourier transforms, and random number generation, as well as efficient numerical operations on multidimensional arrays and matrices.
● **NetworkX :** used to create, manipulate, and study the structure, dynamics, and functions of complex graph networks.
● **operator :** It is used for providing mathematical operations**.**
● **prettytable :** It is used to create relational tables**.**
● **sklearn.metrics :** It is used in Scikit-Learn for various datasets, score functions, and performance metrics.
● **sklearn.model_selection :** It is used to Split arrays or matrices into random train and test subsets.
● **matplotlib.pyplot :** it is used to create 2D graphs and plots.

- We have used the soc-sign-bitcoin-alpha: Bitcoin Alpha web of trust network dataset.
- The dataset has the following properties:
  ● Weighted
  ● Signed
  ● Directed
  ● Temporal
- Network representation in the form of edge list and adjacency matrix.For the edge list, if there is an edge between two nodes,then add the nodes into the edge list. For an adjacency matrix, if there is an edge from node 1 to 2, then the adj_matrix [1][2]=1 else it will be 0. The value for adj_matrix [1][2] is 1 if page 1 links with page 2.

```
Edge List:
[[7188, 1], [430, 1], [3134, 1], [3026, 1], [3010, 1], [804, 1],

Adjacency Matrix:
[[0 1 0 ... 0 1 0]
 [1 0 0 ... 0 1 0]
 [0 1 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 1]
 [1 1 0 ... 0 0 1]
 [0 0 1 ... 1 0 0]]
```

1. Number of Nodes: Using the adjacency list we have iterated over each node and saved it in a set to get the unique nodes.

```
num_nodes = len(set(list(csv_df['source'])+list(csv_df['target'])))
print("Number of Nodes:", num_nodes)
```

```
Number of Nodes: 3783
```

2. Number of Edges: Using the adjacency matrix we have iterated over it and have counted the number of 1's in the matrix to get the number of edges.

```
num_edges = len(csv_df)
print("Number of Edges:", num_edges)
```

```
Number of Edges: 24186
```

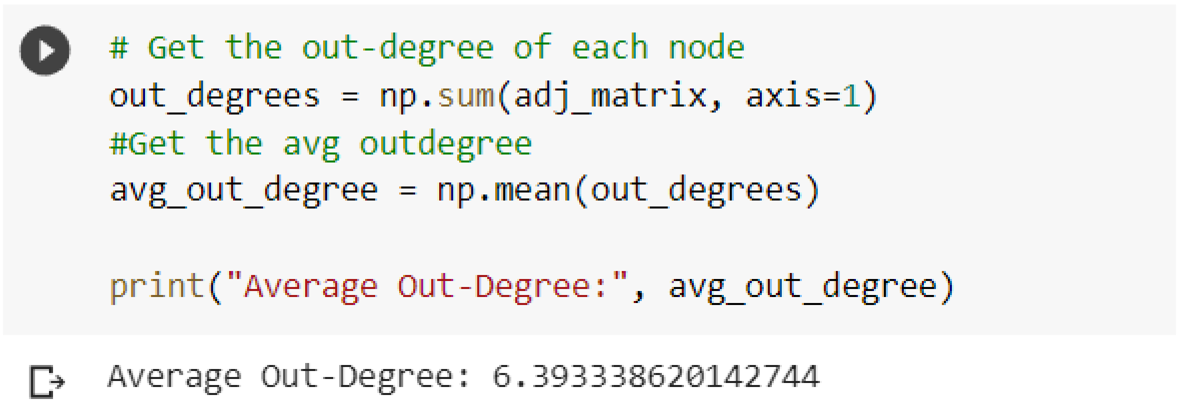
3. Avg In-degree: First we have calculated the in-Degree for each of the nodes and later on sum all the inDegree and divided by the total number of nodes.

```
# Get the in-degree of each node
in_degrees = np.sum(adj_matrix, axis=0)
#Get the avg indegree
avg_in_degree = np.mean(in_degrees)

print("Average In-Degree:", avg_in_degree)
```

```
Average In-Degree: 6.393338620142744
```

4. Avg. Out-Degree: First we have calculated the Out-Degree for each of the nodes and later on sum all the Out-Degree and divided by the total number of nodes.

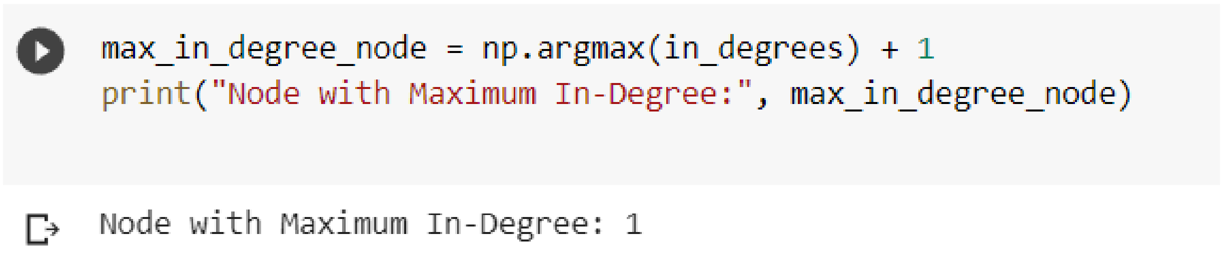


```python
# Get the out-degree of each node
out_degrees = np.sum(adj_matrix, axis=1)
#Get the avg outdegree
avg_out_degree = np.mean(out_degrees)

print("Average Out-Degree:", avg_out_degree)
```

```
Average Out-Degree: 6.393338620142744
```

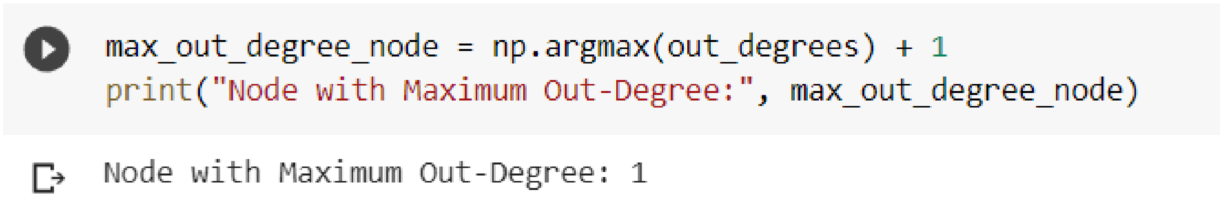
5. Node with Max In-degree: We will check the in-degree of each node and from there we have found the maximum indegree node.



```python
max_in_degree_node = np.argmax(in_degrees) + 1
print("Node with Maximum In-Degree:", max_in_degree_node)
```

```
Node with Maximum In-Degree: 1
```

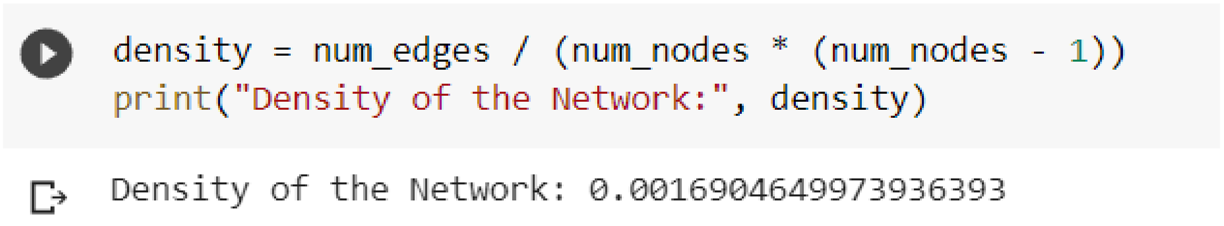
6. Node with Max Out-degree: We will check the out-degree of each node and from there we have found the maximum indegree node.



```python
max_out_degree_node = np.argmax(out_degrees) + 1
print("Node with Maximum Out-Degree:", max_out_degree_node)
```

```
Node with Maximum Out-Degree: 1
```

7. The density of the network: The density of the network is computed by the number of nodes divided by the total number of possible nodes in the network.

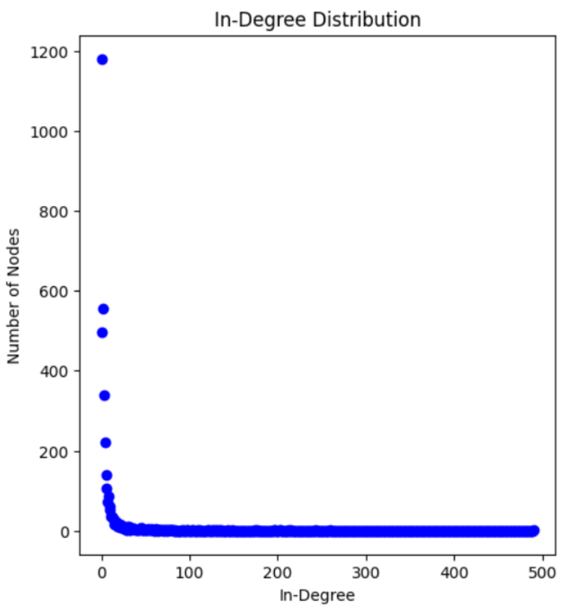


```python
density = num_edges / (num_nodes * (num_nodes - 1))
print("Density of the Network:", density)
```

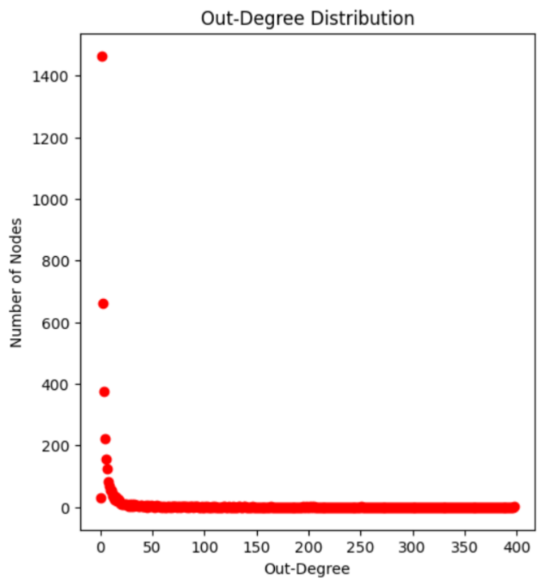
```
Density of the Network: 0.0016904649973936393
```

- **Plot degree distribution of the network**:

  - **Indegree distribution of the network:**

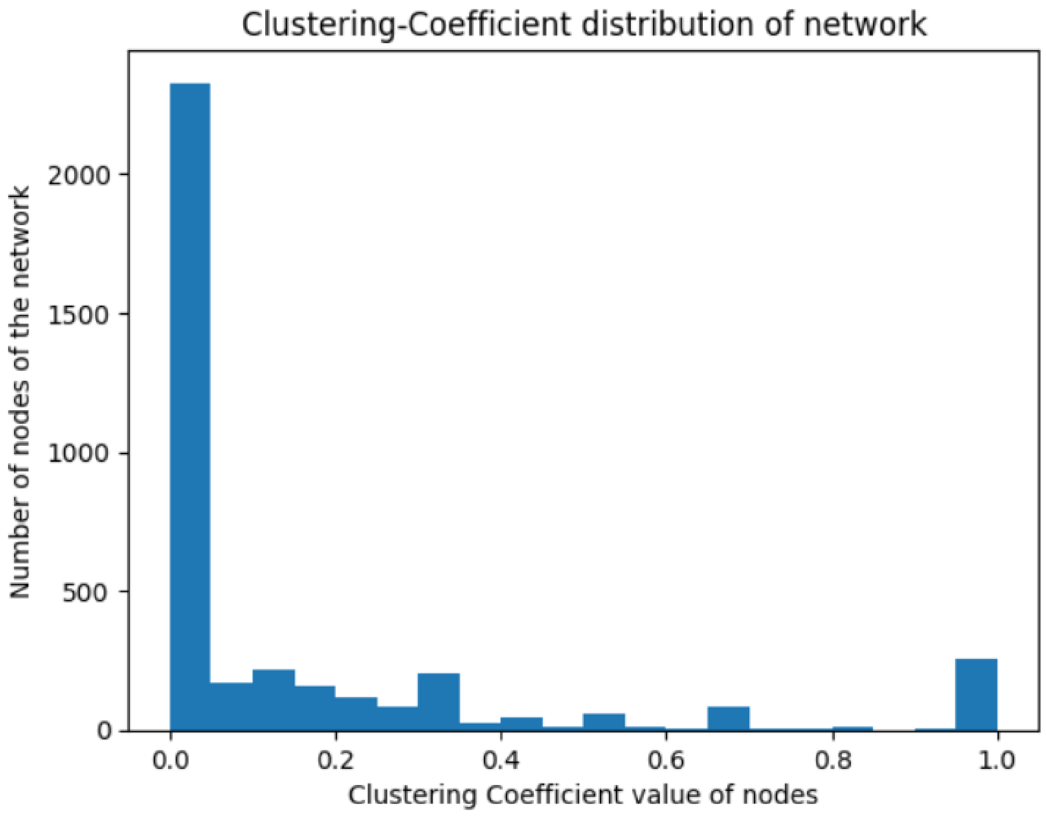


- **Outdegree distribution of the network:**



- **The local clustering coefficient of each node and plot of the clustering-coefficient distribution of the network:**
  To calculate the local clustering coefficient, we have gone through each of the nodes, and after that, we found all the neighbors of that node. After that, for all these neighbors we have seen how many links are there between each of its neighbors. Now we divide this value by the total possible number of links between these neighbors to get the local clustering coefficient for each node.

```
local clusturing coefficient of each node : [0.008176904674506031, 0.05332376853180296
```



❖ **Question 2:**

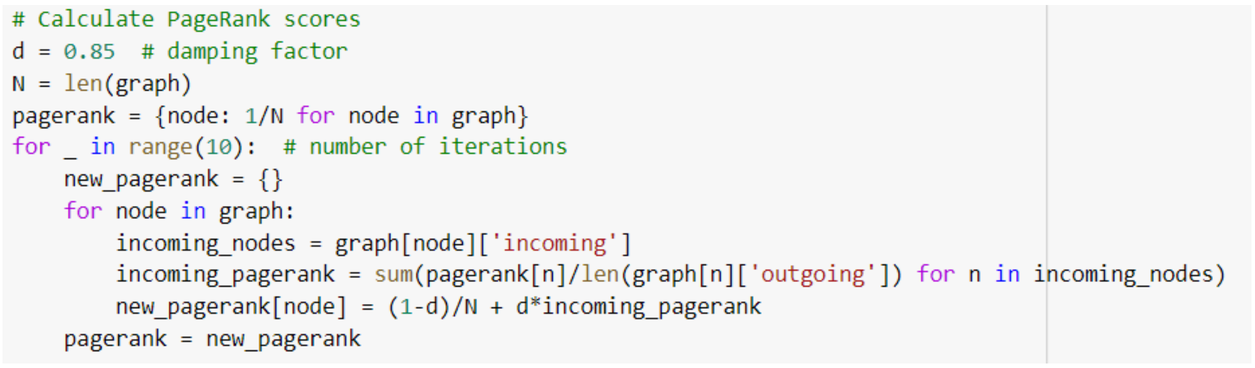
- **Necessary Libraries:**

● **pandas:** It offers capabilities for cleaning, combining, and reshaping data as well as data structures for effectively storing and handling huge datasets.
● **string:** For working with text data present in dataset files.
● **np:** used for working with arrays. It also has functions for working in the domain of linear algebra, fourier transform, and matrices.
● **Networkx:** used to create, manipulate, and study the structure, dynamics, and functions of complex graph networks.

Database: soc-sign-bitcoin-alpha

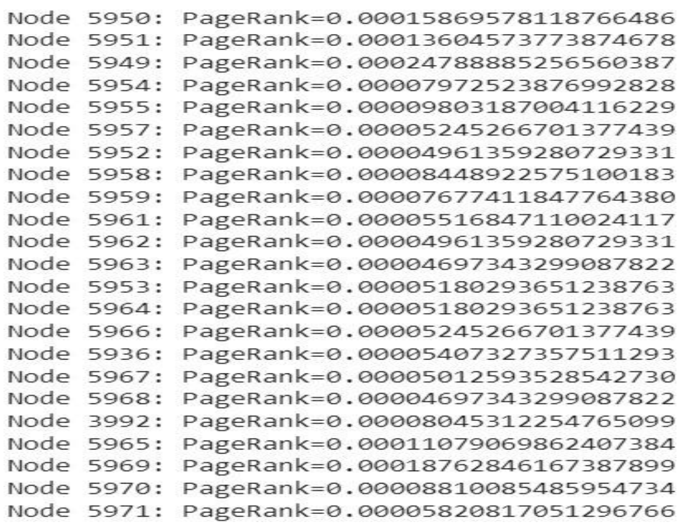
The graph is already constructed in question 1. Pagerank,Hubs and authority scores were computed using networkx library.

1. **Pagerank Score:** Pagerank is computed using the following pagerank method.

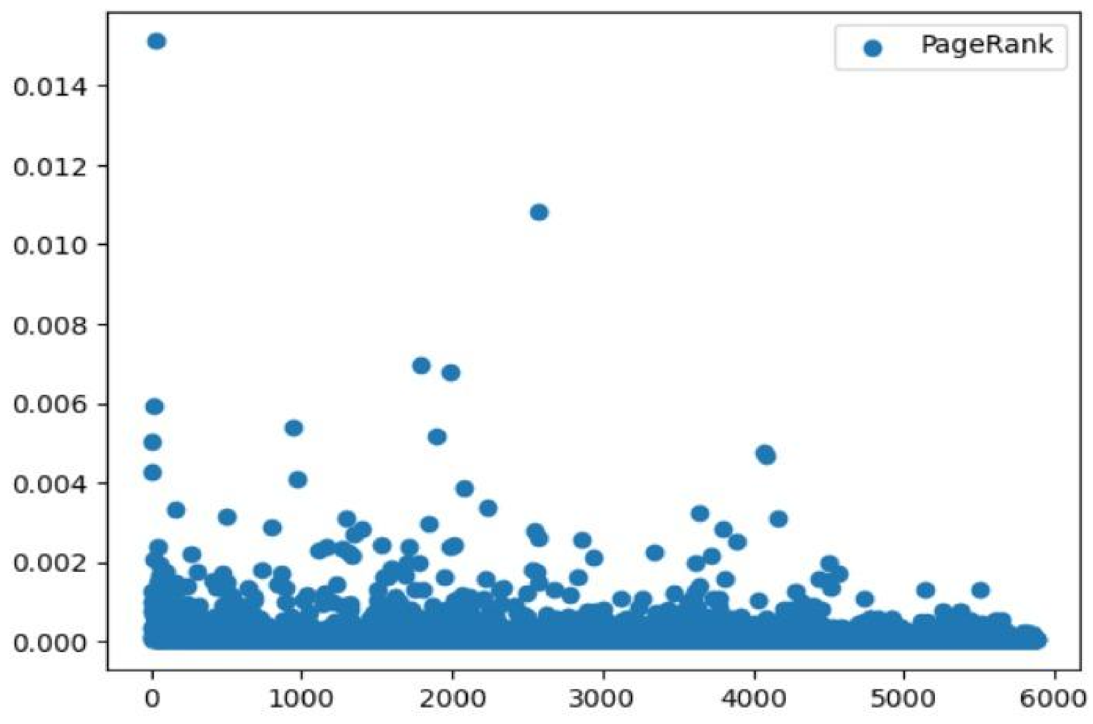


```
# Calculate PageRank scores
d = 0.85  # damping factor
N = len(graph)
pagerank = {node: 1/N for node in graph}
for _ in range(10):  # number of iterations
    new_pagerank = {}
    for node in graph:
        incoming_nodes = graph[node]['incoming']
        incoming_pagerank = sum(pagerank[n]/len(graph[n]['outgoing']) for n in incoming_nodes)
        new_pagerank[node] = (1-d)/N + d*incoming_pagerank
    pagerank = new_pagerank
```

Output: We got the following results



```
Node 5950: PageRank=0.00015869578118766486
Node 5951: PageRank=0.00013604573773874678
Node 5949: PageRank=0.00024788885256560387
Node 5954: PageRank=0.00007972523876992828
Node 5955: PageRank=0.00009803187004116229
Node 5957: PageRank=0.00005245266701377439
Node 5952: PageRank=0.00004961359280729331
Node 5958: PageRank=0.00008448922575100183
Node 5959: PageRank=0.00007677411847764380
Node 5961: PageRank=0.00005516847110024117
Node 5962: PageRank=0.00004961359280729331
Node 5963: PageRank=0.00004697343299087822
Node 5953: PageRank=0.00005180293651238763
Node 5964: PageRank=0.00005180293651238763
Node 5966: PageRank=0.00005245266701377439
Node 5936: PageRank=0.00005407327357511293
Node 5967: PageRank=0.00005012593528542730
Node 5968: PageRank=0.00004697343299087822
Node 3992: PageRank=0.00008045312254765099
Node 5965: PageRank=0.00011079069862407384
Node 5969: PageRank=0.00018762846167387899
Node 5970: PageRank=0.00008810085485954734
Node 5971: PageRank=0.00005820817051296766
```



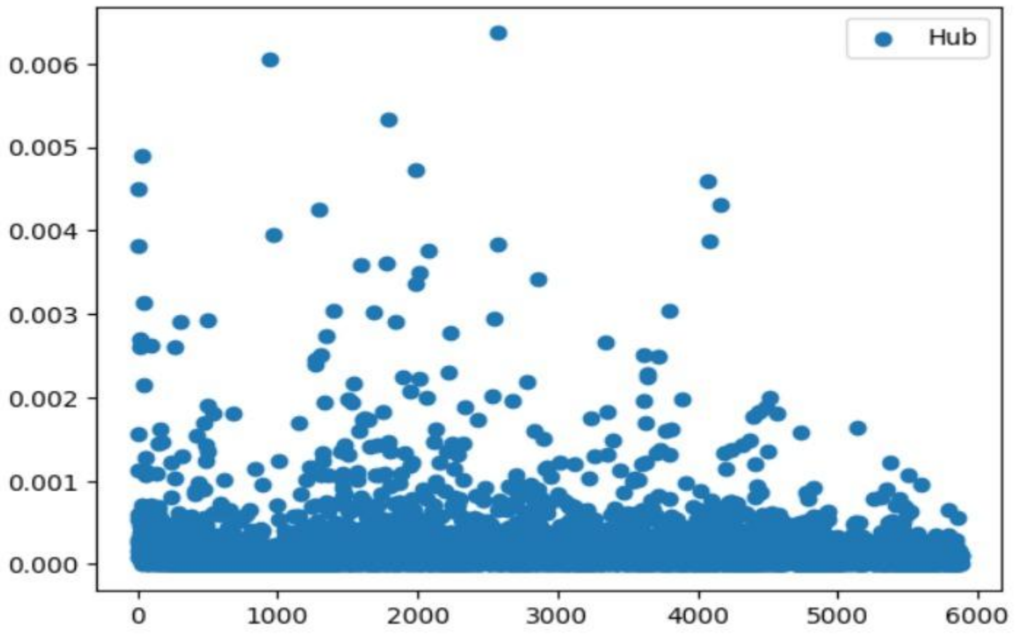
2. **Authority and Hub Score:** Authority and Hub is calculated using following methods.

```
# Calculate HITS scores
authority = {node: 1 for node in graph}
hub = {node: 1 for node in graph}
for _ in range(10):  # number of iterations
    new_authority = {}
    new_hub = {}
    for node in graph:
        incoming_nodes = graph[node]['incoming']
        outgoing_nodes = graph[node]['outgoing']
        incoming_authority = sum(hub[n] for n in incoming_nodes)
        outgoing_hub = sum(authority[n] for n in outgoing_nodes)
        new_authority[node] = incoming_authority
        new_hub[node] = outgoing_hub
```

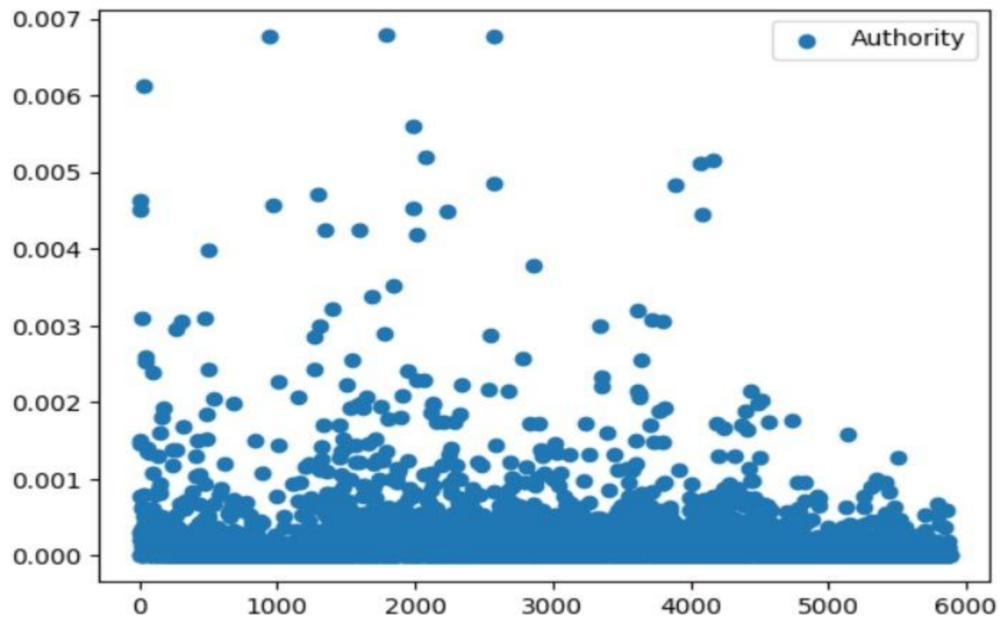
Output: We got the following results.

```
Authority=0.00000995451657331710, Hub=0.00001347497643316448
Authority=0.00001027513762617550, Hub=0.00001365429449910221
Authority=0.00001035987237354906, Hub=0.00001369419898827575
Authority=0.00000032408027138010, Hub=0.00000018373579324683
Authority=0.00013297304466135002, Hub=0.00001368344808138185
Authority=0.00003934606302128441, Hub=0.00002921068086894686
Authority=0.00000000000000000000, Hub=0.00001327934104556332
Authority=0.00001035137576190812, Hub=0.00001594674127483354
Authority=0.00000032935268914688, Hub=0.00000019121766029217
Authority=0.00001588176490512493, Hub=0.00001178485080790725
Authority=0.00000962170466564853, Hub=0.00001327934104556335
Authority=0.00008098108232379759, Hub=0.00008360046959034000
Authority=0.00000000000000000000, Hub=0.00011305265775974630
Authority=0.00000000000000000000, Hub=0.00011305265775974625
Authority=0.00003934606302128441, Hub=0.00002921068086894692
Authority=0.00001686813994442784, Hub=0.00001459741794720119
Authority=0.00002172213347633493, Hub=0.00001842912412586740
Authority=0.00008098108232379759, Hub=0.00008360046959033982
Authority=0.00013084124939510532, Hub=0.00024102566563400442
Authority=0.00036830431234165505, Hub=0.00029898688974417542
Authority=0.00007681727146242285, Hub=0.00018590445907719209
Authority=0.00000447110003103422, Hub=0.00000141785713080625
Authority=0.00011242625037251296, Hub=0.00009649463407036770
Authority=0.00000000000000000000, Hub=0.00000207511094504890
```

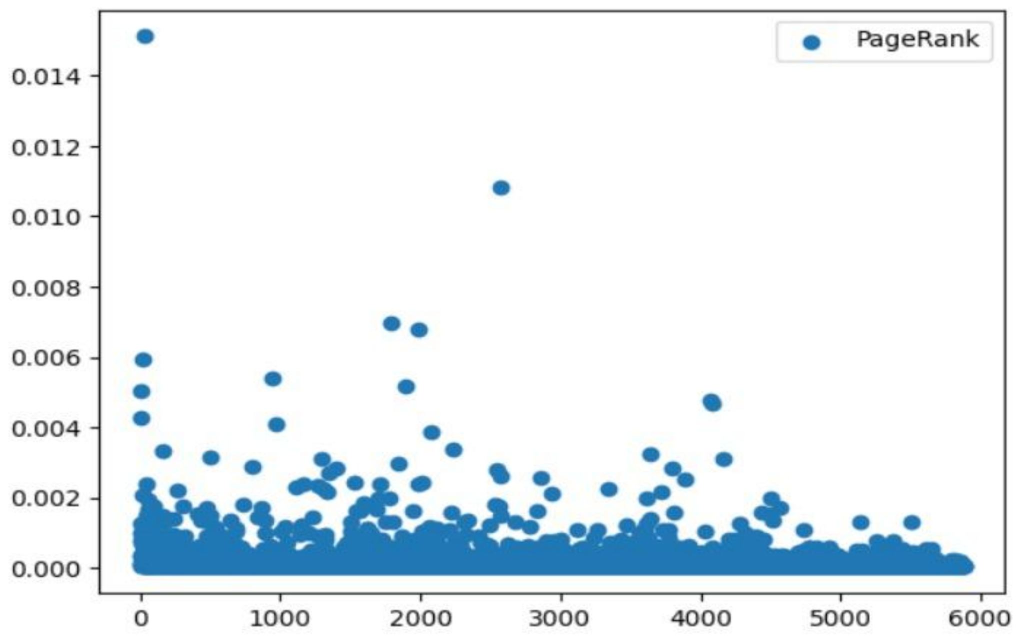
- This is the Scatter plot for Hub Scores:



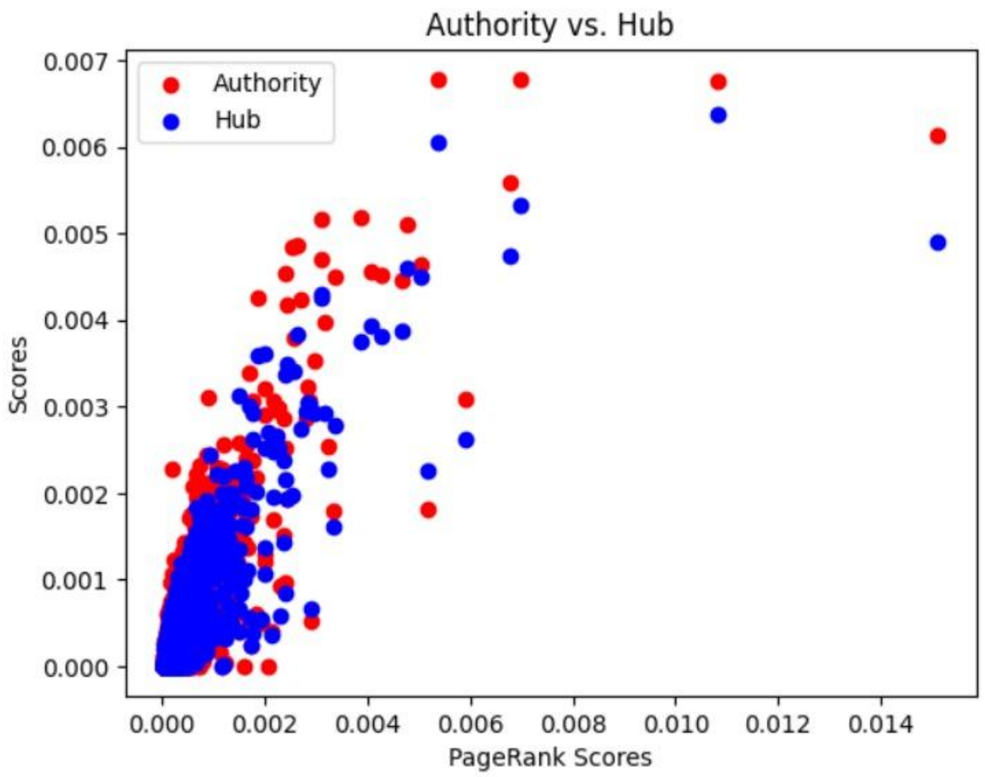
- This is the Scatter plot for Authority Scores:



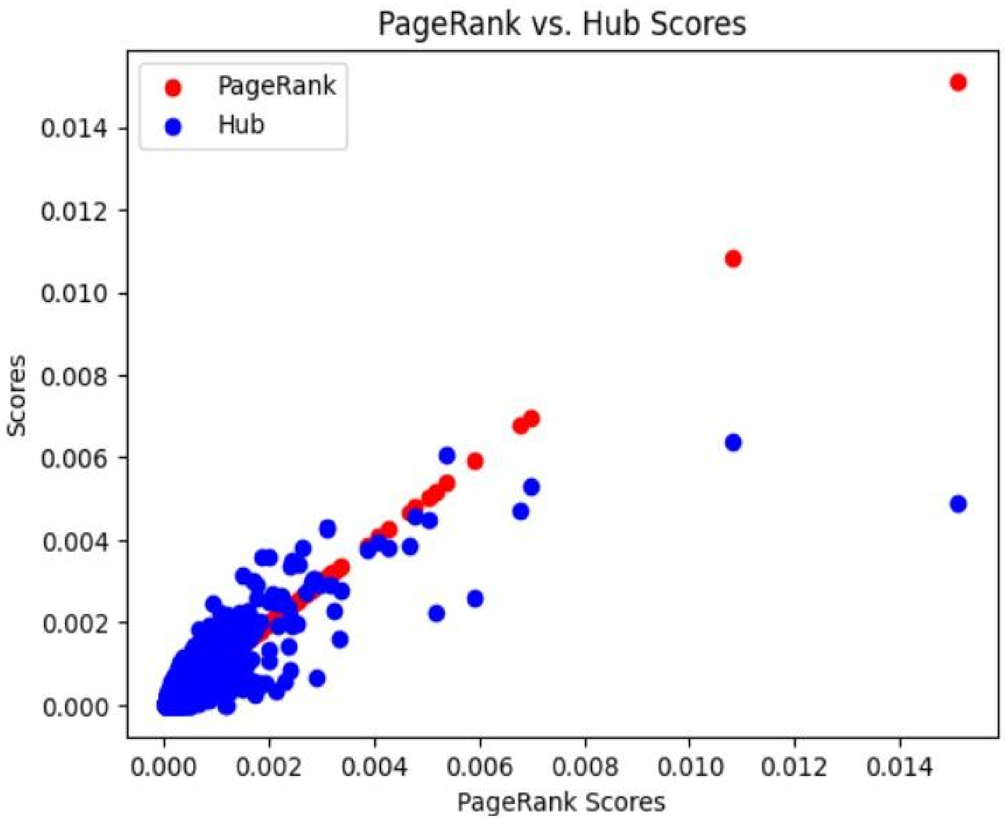
- This is the Scatter plot for PageRank Scores:



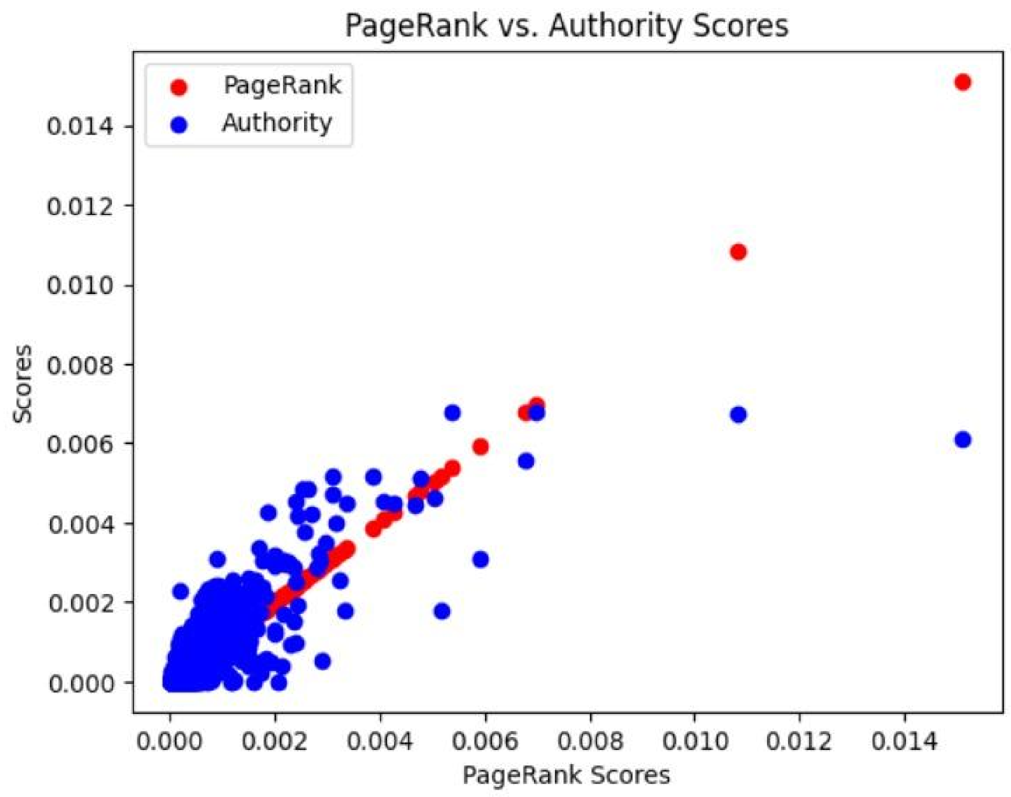
3. **Comparison:**
   - **Authority vs Hub:**



   - **PageRank vs Hub:**

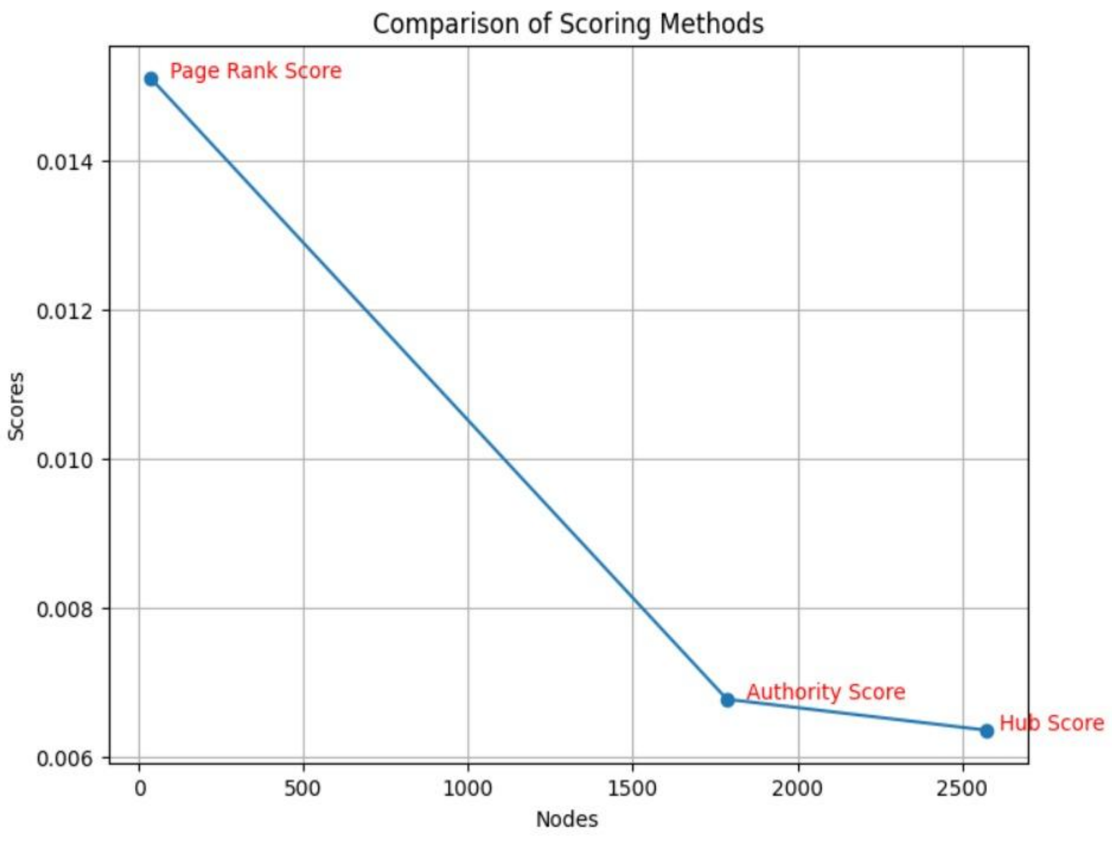


- **Pagerank vs Authority:**



- **Comparison and Authority, Hub and Pagerank scores:**

This is the comparison of Scoring Methods and this is for the highest ranked node and highest score.


Comparison of Scoring Methods
Page Rank Score
Authority Score
Hub Score
Scores
Nodes