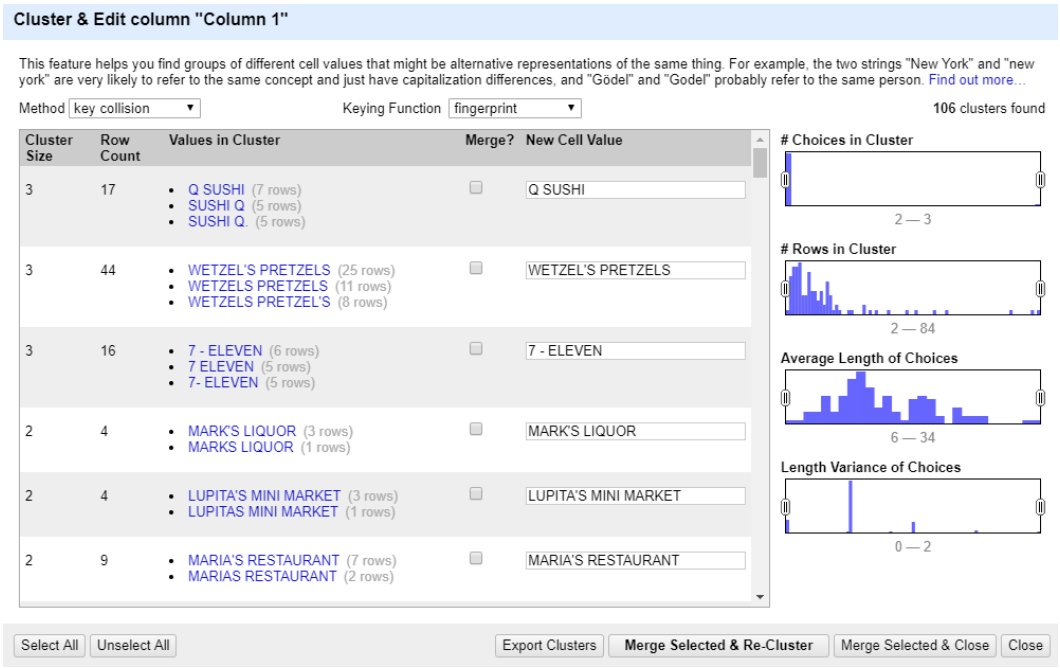
Homework #4: Clustering & Link Analysis using Spark

Due: November 22, Friday
100 points

1. [Clustering, 50 points] Implement the "fingerprint" key collision method in the project using Spark. Name your script fingerprint.py. The process of generating keys should be the same as that given in OpenRefine (you may reuse their code).
https://github.com/OpenRefine/OpenRefine/wiki/Clustering-In-Depth#fingerprint
https://github.com/OpenRefine/OpenRefine/blob/master/main/src/com/google/refine/clustering/binning/FingerprintKeyer.java
This is an example of the key collision result by OpenRefine.



Your implementation should produce the same output as OpenRefine.
Make sure the computation of keys and also the bucketing process are done in parallel.

Execution format: spark-submit fingerprint.py <input-file>

The output should be in a text file named as p1.txt. In the output file, every cluster will be in a single line with key at the beginning. For example, the output for this figure should be:

Q SUSHI:Q SUSHI(7),SUSHI Q(5),SUSHI Q.(5)
PRETZELS WETZELS:WETZEL'S PRETZELS(25),WETZELS PRETZELS(11),WETZELS PRETZEL'S(8)
….

Values in cluster are ranked in descending by its number of items. If there are some values with the same numbers, the result should be same with the result by python default sort. For example, there are 7 items named as Q SUSHI, 5 items as SUSHI Q and 5 items as SUSHI Q., so we will output it as:

Q SUSHI:Q SUSHI(7),SUSHI Q(5),SUSHI Q.(5)

Python default sort means the sort function without any other argument. For example:

```
stringlist = [“SUSHI Q.”, ”SUSHI Q”]
stringlist.sort()
```

then the stringlist will be ['SUSHI Q', 'SUSHI Q.']

2. [HITS, 50 points] Implement the “HITS” algorithm for computing authorities and hubs in Spark. Name your script hits.py. You may take the pagerank.py seen in class as the starting point. The algorithm should start with initial hub vector of all 1’s and use mutual recursion to compute authority and hub scores for a specified number of iterations. The scores of an authority/hub vector should be normalized so that the largest score is 1.

   Execution format: spark-submit hits.py <graph-file> <# of iterations>

   Note that <graph-file> is a text file with each line representing a directed edge in the graph, similar to the data file for pagerank computation. There is an example in graph.txt.

The output should be in a text file named as p2.txt with two lines. The first line will be the result of h and the second line will be the result of a after <# of iterations>. For example:

1 0.36 0 0.72 0

0.21 1 1 0.79 0

Please upload a zip or tar file which contains all your code and name it as FirstName_LastName_hw4.tar or FirstName_LastName_hw4.zip