ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

SCHOOL OF BUSINESS

DEPARTMENT OF MANAGEMENT SCIENCE & TECHNOLOGY

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

ACADEMIC YEAR OF 2021 – 2022

ADVANCES TOPICS IN DATA ENGINEERING

ASSIGNMENT II

KONSTANTINOS NINAS

f2822108

SUPERVISING INSTRUCTOR

GIORGOS ALEXIOU

Table of Contents

Task A – Conduct Token Blocking on the Dataset........ 3

Task B - Calculate the total number of comparisons needed to examine all the token blocks........ 3

Task C – Create a Meta-Blocking Graph........ 4

Task D – Create a function that calculates the Jaccard similarity between the titles of the publishes of any two authors........ 4

Task A – Conduct Token Blocking on the Dataset

We belong in a data engineering team in the company X, and we received a dataset from a customer, requesting from us to conduct an Entity-Resolution process. Entity-Resolution is the process with which it is possible to identify and aggregate the different entity profiles/records that actually describe the same real-world object. The dataset contains names of authors, the name of the venues, the year they were conducted and the title of their publish. In order to tackle this task, we will conduct token blocking on the dataset. Specifically, we will take each one of the above attributes recorded and tokenize it for each record. Tokenization is the process in which meaningful data (such as the name of the venue for an author) are divided into multiple strings (one for each word) with no special meaning. Before we conduct the tokenization process, we will convert all characters to lower, so they will all be consistent with each other, since some tokens may be the same with the exception of some capital characters. In addition, we will also remove all unnecessary symbols, such as ‘,’,’.’,’$’ and more for similar reasons. Once these actions are completed, we will create a dictionary using Python scripts, and will store the tokens as keys, and the ids of the authors that have each token in any of their attributes as values. To reduce the storage space needed we also removed all keys that had only one author id in their values. We also removed keys with null values since we believe that they do not assist towards the identification and aggregation of different author profiles that describe one real-world individual. Authors with many common tokens will be further examined to check whether they refer to the same person, while those with little to no common tokens will not be examined, since it is almost certain that they do not refer to the same individual.

Task B - Calculate the total number of comparisons needed to examine all the token blocks

Next, we are asked to calculate the number of total comparisons needed to examine all the ids of the authors in all the blocks created in the first task. To tackle this task, for each key we counted the number of values it contained and applied the following equation: $temp\_sum(key) = \frac{n*(n-1)}{2}$, where n is the number of values for each key-token. We use that equation because, for example, if we compare the author with id1 and the one with id2 in one block, it is not needed to compare the author id2 with the one with id1 in the same block. We sum the temporary sums of each key-block in a new variable. When we have iterated through all keys, that variable will contain the number of all comparisons needed to go through all author ids in all token blocks. It was found that 2.707.929.957 comparisons are needed for that task.

Task C – Create a Meta-Blocking Graph

Following, we were asked to create a Meta-Blocking Graph based on the blocks created in Task A. The first step for this process is to create the nodes and the edges of the graph. At first, an empty list had to be created to store the nodes and the edges that will be later added to the graph. Then, we iteratively examined each block, and created added nodes in the list of nodes when a new id was identified. Additionally, all pairs of ids were added as edges in the edge list. Once all the edges were created, the weights of each edge were calculated. For example, if there are 4 edges between authors with id1 and id2, which means that they appear together 4 different blocks, we create one edge between them with a weight equal to 4). Then, we set as a threshold a weight equal to 2 for the edges we wanted to keep in the graph, so we prune all edges that had a weight smaller than 2. We did that because we considered authors with only one common token are highly unlikely to refer to the same real-world individual. Then, to calculate the new number of comparisons needed to examine the pairs of entities in the graph, we just had to count the number of edges that remained in the graph. It should be mentioned that due to the high volume of the data, it was impossible to perform all the above calculations on all the blocks created from the first task. Instead, they were performed on a very small sample of the blocks (10 blocks), to also verify the functionality of the python scripts that are provided.

Task D – Create a function that calculates the Jaccard similarity between the titles of the publishes of any two authors

Finally, we were asked to construct a function that will calculate the Jaccard similarity between the tokens of the title of the publishes of any two authors. First, we needed to construct a function that would calculate the Jaccard similarity between two sets. In specific, the Jaccard similarity is the division between the number of the common elements and the number of the union of all the elements between two lists. To calculate the Jaccard similarity between two titles, both titles should be tokenized, which means we would have to keep all distinct strings of each title and store them as elements in lists (one for each title). Then the tokens were used as elements in the lists to calculate the Jaccard similarity between the two titles.