Task 1C – Report on Data Linkage Project

Task 1A

In task 1A, we were given small datasets since and were asked to compare every record-A to a possible match in record-G (Amazon and Google). The main columns for both tasks were id, name, description, manufacturer (4 strings) and price(float). The description column in both datasets, and the manufacturer column in google had many null values and were unreliable to help the comparison of records.

To compare and score the records, I used the .partialratio and extractBests functions from the Fuzzywuzzy library. With no required preprocessing, the name was the chosen column to score since there were only three columns to experiment with, and the name stood out as a strong benefactor to the accuracy of the model. Consequently, an initial threshold of 50 out of a possible 100 was selected since those records with less than half of the name matching would have a very low probability of being a true match. The threshold was then raised to 65 with a focus to increase the precision and recall to over 90% for the model.

With the current metrics, the model does well for the data given and uses a simple scoring algorithm to achieve its goal and which works well with new data. However, it could be improved by comparing price and description – not to match but – to ensure that the match that has occurred is accurate and justified. This can be done by applying weight on name, description, and price, with the majority of the weight on name since it is the key factor. This allows the name to be the key deciding factor while the description and price act as the bias towards which helps when the score is around the threshold.

Task 1B

The blocking method uses bigrams (A sequence of two adjacent tokens from a string) for every word in the name column of the record and assigns it to a block of every combination possible with two words. Eventually, if products are a match between the two datasets, they must at least have two similar words to fit a similar blocking key. The name column was chosen again since it has the most relevant keywords with the least amount of words to make blocks out of.

The complexity of the blocking algorithm is linear since it takes a unit of time to create the block of a record based on its name and goes through n records. The records do not compare or assign themselves to pre-made blocks but simply create blocks on the go with the words in their name. Other products, if similar, would also contain the same words to match the same block. The PC and RR are both above 90% which shows that technique provides to be useful to compare and connect records to blocks. However, we could improve the algorithm by extracting features such as keywords like ‘Microsoft’ or ‘Apple’ and other important or recurring keywords to optimize the time to run the code and minimize the number of blocks. This can be done by using libraries such as TFidfVectorizer (Term Frequency Inverse Document Frequency) from sklearn to help provide a list of key vectors that are useful to facilitate the creation of meaningful blocks.