Spark! Heyrick Research Fuzzy Matching Deliverable 2

Team: Suzy Kirch, Yang Hu, Chengyang He, Haoran Kang

Our task is to create a model that can take two datasets and find the matches between. To do so, we need matching criteria, that is, we need match thresholds for name, address, name and address combined, and phone number.

Our preliminary results were gathered using Rubmaps and Google Places data for the state of Massachusetts. The dataset we have submitted is an excel file called above85.xlsx, which lists the matches based on our preliminary match criteria (see page 2).
The columns are as follows:

- index
- location_hash: the unique ID for the business in the Rubmaps dataset
- googleID: short for googlePlaceID, the unique ID for the business in the Google Places dataset
- rubmap_ad: the business' name and address, separated by a comma, as stated in the Rubmaps dataset
- google_ad: the business' name and address, separated by a comma, as stated in the Google Places dataset
- name_score: the calculated score of name comparison
- addr_score: the calculated score of address comparison
- name_addr_score: the calculated score of the comparison of the combination of name and address
- g_phone: the business' phone number, as stated in the Google Places dataset
- r_phone: the business' phone number, as stated in the Rubmaps dataset
- phone_score: the calculated score of phone comparison

All comparison scores were calculated with fuzz.token_set_ratio in the fuzzywuzzy python library. 100 is a perfect match. 0 is a complete non-match.

We specifically chose fuzzywuzzy for a couple of reasons. One reason is that it seemed easier to use, which meant it was a good starting place. More importantly, though, from our research, it appeared as though the other library of choice, spaCy, was really only good at identifying identical matches, which is not what we need for this project.

We settled on fuzz.token_set_ratio after trying several different fuzzy matching score calculators. We found that the results of fuzz.token_set_ratio was the closest match to how we would score matches and what we would consider to be matches.

In our trials of fuzz.token_set_ratio, we quickly realized that an address match would not be sufficient, as you could have a shared address, but a different suite in the building, for example, and a completely different business name, and these would disqualify the pair from being a match. We are therefore comparing name and address together, and verifying the match with individually matching name, address, and phone.

After investigating the matches and non-matches, we have identified the following to be our preliminary match criteria. This will be solidified more over the coming couple of weeks. We want our matches to err on the side of caution, so in short, we are trying to make the number of false positives as small as possible.

- First and foremost: name_addr_score >= 85
- Other matching criteria: The business must pass at least 2 additional matches following this criteria.
    - addr_score >= 90: Address is secondary. Anything other than a slightly different number, apt, suit, etc discrepancies will be non-matches.
    - name_score >= 90: Name is tertiary match.
    - phone_score = 100: Only a perfect phone match will do. A phone number match is the weakest link, since phones can be shared across businesses.