CS 4400X Final Project

1 GitHub Repository Code

The code for this solution is at https://github.com/YBLim13/CS4400X-ylim62-FinalProject

2 Solution Outline

The solution has five steps: (1) data reading, (2) blocking, (3) feature engineering, (4) model training and (5) generating the output .csv file.

2.1 Data Reading

In this step, we read in the data from the left table, the right table, and the training set

2.2 Blocking

In this step, we block the data primarily on the attribute "brand" and then secondarily on the attribute "modelno" to create a candidate set of pairs. When blocking on "brand," we select pairs where one (or both) of the brands is an empty string, where both brands are equal, where one brand is a substring of the other brand (ignoring spaces and non-alphanumeric characters), or where the brand of one is a substring of the title of the other. The intuition here (going down the list) is that just because one or both brands are empty does not mean that they are not the same entity, that two products with the same brand have a higher chance of being the same entity, that sometimes the recorded brand has more info than necessary (e.g. "post-it (3m)" and "3m"), and that even if the recorded brands are not the same the title will often reference the actual brand name. When blocking on "modelno," we select the pairs in the same way as for "brand" except that we do not consider the pair if one or more modelno are empty strings. The purpose of this secondary blocking is that we are trying to find the true pairs where even with different recorded brands the modelno is the same. The intuition here is that given a true pair with different recorded brands, chances are relatively high that their model numbers are going to be the same. But if the recorded brands are different (they are assumed to not be empty strings since those would already be considered in the primary blocking) and both modelno are empty strings, chances are that they are different entities. (The intuition for the other selection criteria remain the same.) Blocking in this manner reduces the number of pairs from 56,376,996 to 3,478,450. It is admittedly still a lot, but it gets rid of nearly 94% of all of the pairs.

2.3 Feature Engineering

For each pair in the candidate set, we generate a feature vector of 15 values by obtaining the Jaccard Similarity, Levenshtein Distance (normalized), and Ratcliff-Obershelp Similarity values of the pair on its five attributes. The intuition there is that we should create a feature vector with values that are (respectively) token-based,

edit-distance-based, and sequence-based. In this way, we obtain a feature matrix X_c for the candidate set. We do the same to the pairs in the training set to obtain the feature matrix X_t. The labels for the training set is denoted as y_t.

2.4 Model Training

We use three different classifiers to train three different models: a Random Forest classifier, a Gaussian Naive Bayes classifier, and a K-Neighbors classifier. We train each model on (X_t, y_t). (In the Random Forest classifier, since the number of non-matches is much more than the number of matches in the training set, we also set class_weight="balanced" to handle this training data imbalance problem.) We perform three predictions on X_c based on each of the models to get three predicted labels y_{c1}, y_{c2}, and y_{c3} for the candidate set. From these three predicted labels, we get a final predicted label y_{cf}, where the prediction is determined by the majority of the three intermediate predicted labels.

2.5 Generating the Output .csv File

The pairs in the candidate set with $y_{cf} = 1$ are our predicted matching pairs M. We remove the matching pairs already in the training set from M to obtain $M-$. Finally, we save $M-$ to output.csv