Radius Collider Submission - The Approximators

Myles Scolnick, Shane Barratt and Alexander Danilychev Jr

I. INTRODUCTION

The goal of this project was to determine the best North American Industry Classification System (NAICS) code, for a business based on its name, address, description, and website. The North American Industry Classification System (NAICS) is the standard used by Federal statistical agencies in classifying business establishments. For example, a Safeway would be classified as the code 445110 (Supermarkets and Other Grocery, except Grocery). To solve this problem, we incorporated several techniques from a variety of fields: text cleaning, data augmentation, tf-idf, wordnet, neural word embeddings and cross-validation. A pure supervised learner would be tough to model/train as there are less training data points than classes, so we went with what one could call a 'topic modeling' and 'rule based' approach.

II. DATA

To increase the amount of data on businesses and NAICS codes, we aggregated some data from public APIs including Google Places business type, NAICS code titles/descriptions, and NAICS code freqeuncies. The respective scripts are listed in List 1 in the appendix. We also transformed the data to normalize the text for comparisons: stripped stop words, lemmatized, lowercased, tokenized. We also enhanced the richness of the text by adding synonyms obtained through the WordNet model [3].

III. WEB APP

A. Assisted Hand Classifier

To ease the painful process of hand-classification, we built a web application in the flask framework for python that has a nice user interface for classifying businesses. This allowed classifications to be automatically written to a file. It greatly sped up hand-classifying 1,000 businesses. Refer to the README/github for pictures.

B. Database View and Code Comparison

To further understand the structure of the data and the quality of our classifier, we also built an additional endpoint to visualize some of the classifications the algorithm made with respect to our hand classifications. It allows us to discern what needs to be improved in the algorithm. Refer to the README/github for pictures.

IV. The Algorithm

Let the set of businesses be denoted by $\{b_1, b_2, \ldots, b_n\}$ and the set of NAICS codes by $\{c_1, c_2, \ldots, c_m\}$. The algorithm attempts to construct a similarity matrix $\mathbf{S}$ of dimension nxm where $\mathbf{S}_{i,j}$ denotes the similarity between business i and NAICS code j. The classification for business (row) i is $\arg\max_{1 \leq j \leq m} \mathbf{S}_{i,j}$, the most similar NAICS code to that business. To construct this matrix, we first construct 9 similarity matrices, $\{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_9\}$, derived from different similarity measures between a business and a NAICS code. There are 4 ways to pair text related to the business the business (B) and NAICS (N) $\rightarrow$ {(B.name, B.description) X (N.name, N.description)}. Therefore, the two similarity methods, *tf-idf* cosine similarity [2] and *word2vec* cosine similarity [4] as well as a similarity matrix of 2-digit industry priors (independent of the business) constitute the 9 similarity matrices. The final similarity matrix is a convex combination of the preliminaries: $\mathbf{S} = \sum_{i=1}^{9} w_i \mathbf{S}_i$,

where $\sum_{i=1}^{9} w_i = 1$. The weights were selected through random hyper-parameter search [1]. Again, the algorithm classifies businesses based on $f(b_i) = \arg\max_{1 \leq j \leq m} \mathbf{S}_{i,j}$.

As well as this brute-force similarity approach, we also implemented rule-based classification for common phrases that occur in the title or description. For example, any businesss that mentions restaurant is automatically classified as 72251 (Restaurants and other eating places), because there are rarely false positives in this case. On top of this, we also implemented thresholding. If the highest similarity metric between a business and the codes is below a threshold, the algorithm will not classify the business.

The codebase is in python, and used some packages including gensim (word2vec), nltk (nlp) and scikit-learn (tf-idf). Github was also used for version control.

V. CONCLUSION

A. Results

After training, our algorithmic classifier scores about 31% against our hand-classified 1000 businesses.

B. Confidence, Limitations and Unexplored Ideas

- We are not particularly confident in the predictions our algorithm made, as it has lots of false-positives.
- The algorithm did not take into account the hierarchical structure of the NAICS codes. We believe this could help the score by determing the marginal expected value for each additional digit.
- Collecting data was pretty limited, but if possible to gather the approximate amount of revenue and/or costs coming from a business, it would be helpful in narrowing down the correct NAICS industry.

C. Challenges

- The data did not come perfectly formatted and sometimes the title, description or website did match and was a mix of 2 businesses.
- Some of the businesses had names and descriptions in other languages.
- Some businesses have multiple obvious classifications making it difficult to assign a single NAICS code (*e.g.* a bakery which also is a sit-down restaurant).

References

[1] Bergstra, James, and Yoshua Bengio. "Random Search for Hyper-parameter Optimization." *The Journal of Machine Learning Research* 13.1 (2012): 281-305.

[2] Brown, Peter F., et al. "A Statistical Approach to Machine Translation." *Computational Linguistics* 16.2 (1990): 79-85.

[3] Miller, George A., et al. "Introduction to Wordnet: An On-line Lexical Database*." *International Journal of Lexicography* 3.4 (1990): 235-244.

[4] Mikolov, Tomas, et al. "Distributed Representations of Words and Phrases and their Compositionality." *Advances in Neural Information Processing Systems*. 2013.