Case Study: Names Mapping

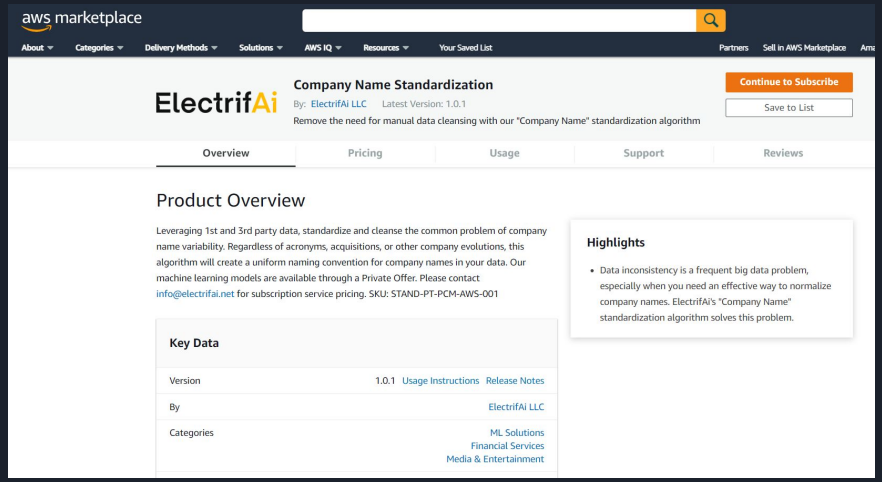
Mohamed MESSALTI

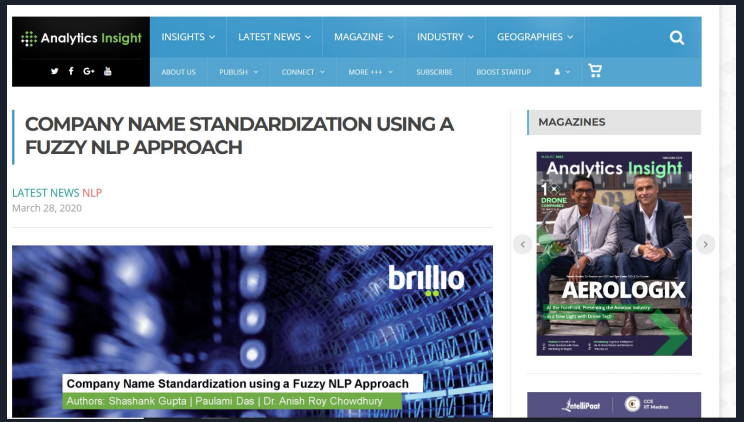
Estimated working time: 1 day

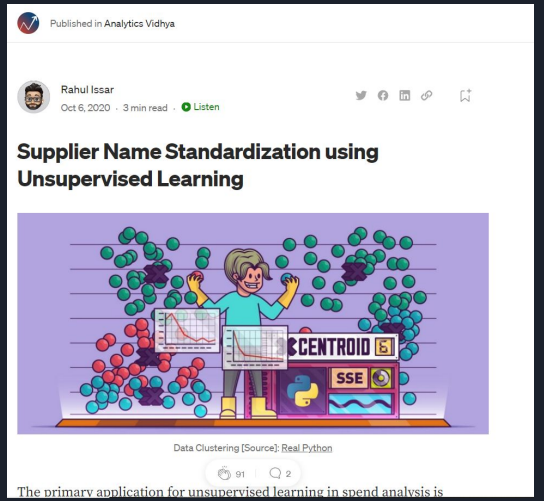
Working time consumed : 2-3 days

The State of the Art

Not being an expert in NLP, the first step of my work was to examine the state of the art and the existing solutions.







The State of the Art

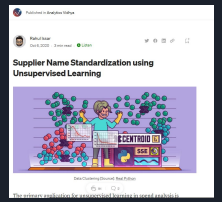
Why Levenshtein Distance? Our use case requires the aggregation of similar names. This metric takes into account the minimum number of single-character changes (insertions, deletions, or substitutions) needed to transform one word into another.

Why clustering with affinity propagation? It does not need to specify the number of clusters.

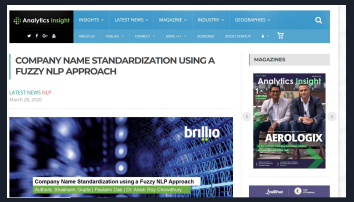
Coding Workflow Overview

My code is partly based on the following 2 articles:

- Supplier Name Standardization using Unsupervised Learning, Rahul Issar, Oct 6, 2020



- Company Name Standardization using a Fuzzy NLP Approach, Shashank Gupta, et al. March 28, 2020



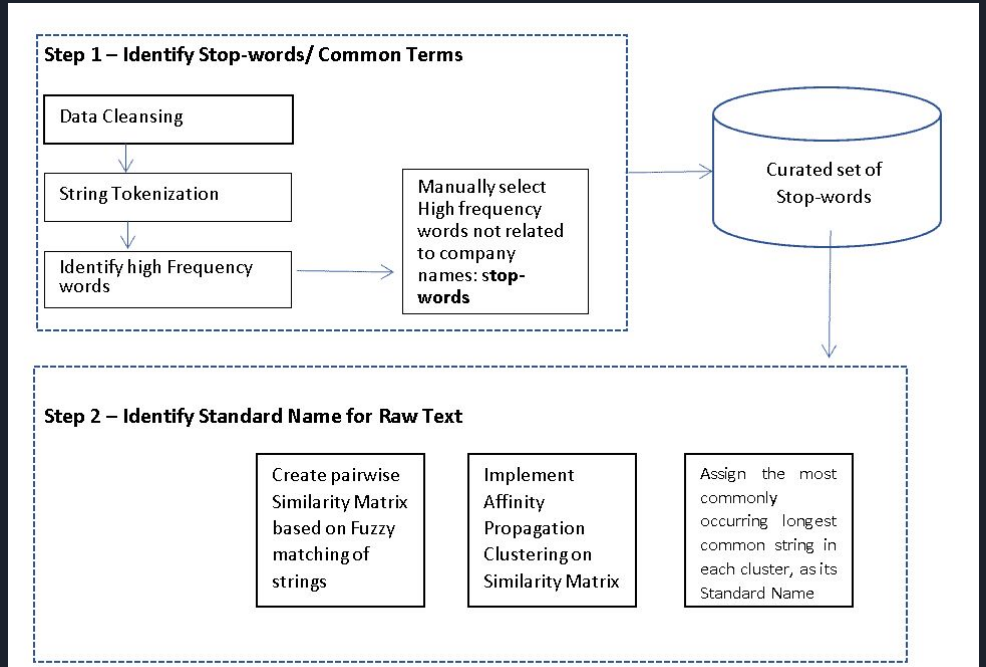


Fig.1 Schematic of two-step solution methodology

Evaluation

Mapped Names

Since the case study focuses on **unsupervised learning, no common metrics are used** to calculate the performance of the algorithm, such as R2 and MSE for regression or accuracy and F1-score for classification.

Labeling the data is time consuming and/or not feasible. Indeed, it would be necessary to define a standard name for the majority of the companies that I do not even know. Suppose I know all the companies, I would have to label 1000 rows to have a test set of only 20% of the 5000 rows and 20000 rows for the 100000 rows dataset.

The **Confidence score** quantifies the confidence with which we can say that the cluster name we identified truly represents the company name for the raw string.

Quite honestly, the best approach for me to evaluate the performance of my algorithm is to go through my result file.

Evaluation

Time complexity

Data set size : Number of rows	Execution time
100	~ 0.350 seconds
1000	~ 42 seconds
4895	~ 1383 seconds = 23 min
100 000	> 3 000 min = 50 hours > 2 days ~ 9 660 min = 161 hours = 6.7 days

x10 ↓
x5 ↓
x20 ↓

↓x120
↓ x33
↓ x132
x420

Processor	Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz
Installed RAM	8,00 GB (7,88 GB usable)

The pairwise similarities computing is O(N2) complex

Critical analysis and suggestions


Super Fast String Matching in Python

Oct 14, 2017

- Some outliers are included in the nearest cluster,
- Names abbreviated may not be mapped to full names (eg. LBP = La Banque Postale)
- special character on companies name are removed
- Subsidiary name which may not be mapped to parent organization name
- If some instances of company names are anomalous, i.e. the name is very different from the actual company name, they may not be identified correctly.
- We have not identified the actual Legal name for the organizations. Cluster names are not always understandable.
- Multilingualism needs a multilingual stopwords?
- If different companies have very similar names, they may be grouped together. For example, "ABC" and "ABC Document Imaging" can be grouped together, even though they are two different companies.
- The confidence score is a cluster- or instance-level indicator. How to evaluate the overall performance of the algorithm?
- How to improve time complexity ? TF-IDF, N-gram, cosine similarity ?