

Company Classification

Technical Documentation

Problem Overview

Here we have a dataset from over 70 thousand companies websites. This dataset is compiled by scrapping the text from the websites of all those companies. Our goal is to divide these companies into some groups where each group would have similar companies. In this way we can approach those companies in the same group with the same kind of business proposals.

We are going to use some clustering technique to classify similar companies in one group. By automating the process of classifying the companies into groups will save a lot of time and the stakeholder can dedicate more time on some meaningful tasks like creating business proposals.

Dataset Description

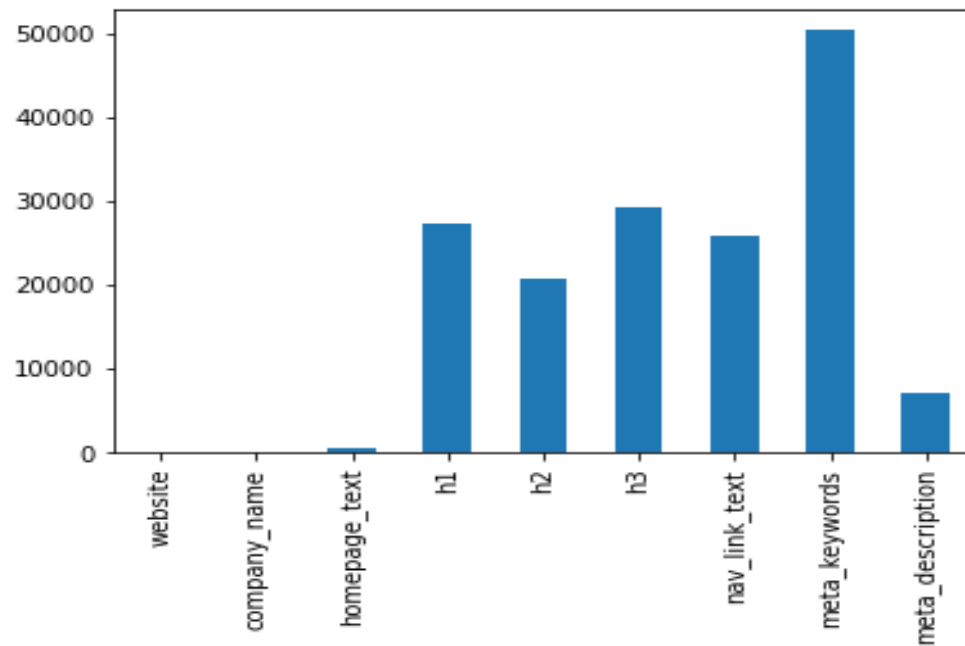
The data set contains information about around 77k companies. Here are the features and their description:

1. **website:** The website of the company/business
2. **company_name:** The company/business name
3. **homepage_text:** Visible homepage text from the company website
4. **h1:** The heading 1 tags from the html of the home page
5. **h2:** The heading 2 tags from the html of the home page
6. **h3:** The heading 3 tags from the html of the home page
7. **navlinktext:** The visible titles of navigation links on the homepage (Ex: Home, Services, Product, About Us, Contact Us)
8. **metakeywords:** The meta keywords in the header of the page html for SEO (More info: https://www.w3schools.com/tags/tag_meta.asp)
9. **metadescription:** The meta description in the header of the page html for SEO (More info: https://www.w3schools.com/tags/tag_meta.asp)

Detailed solution approach

1. Dealing with missing values

There were a good amount of missing values present in the dataset. Initially This is how the null value distribution looked like in the data set:



Missing values distribution in all columns

Some takeaways from the above graph:

- Meta_keywords has most of the values missing, since Meta keywords and meta description represent the similar meaning, we are going to merge these two columns into one column called meta_text and then will analyze them.
- Columns h1, h2, h3 also have a huge amount of missing values, and they also represent the similar meaning. So we decided to merge them into a single column named headings, and after merging them the number of missing values went down to less than 10%.
- Column nav_link_text has a good amount of null values and it was not such an important feature for clustering, so we simply removed it from the data.
- Homepage_text and meta_description are the columns with least number of null values and while looking at the data it was observed that those two columns have the most amount of information so they are gonna be really important while clustering.
- Finally we have columns homepage_text, headings, meta_text which we're gonna perform clustering with.

2. Text preprocessing

This is a crucial step before clustering because this will directly affect how good our clusters are going to be. These are the steps we took for text preprocessing.

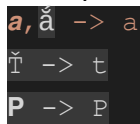
- **Remove Emojis**

These kinds of special characters are going to contribute nothing in the clustering, but they can affect clustering negatively so it's good to remove them. Examples of emojis:



- **Converting visually similar characters to their original form**

Examples:

A dark horizontal bar showing three examples of character conversion: 'a, ä -> a', 't -> t', and 'P -> P'.

If we don't convert those visually similar characters to their base form, they will be considered as special characters and will be removed in further steps. In this way we could have lost some useful information. Hence converting them into the original form is important.

- **Removing the special characters/punctuations**

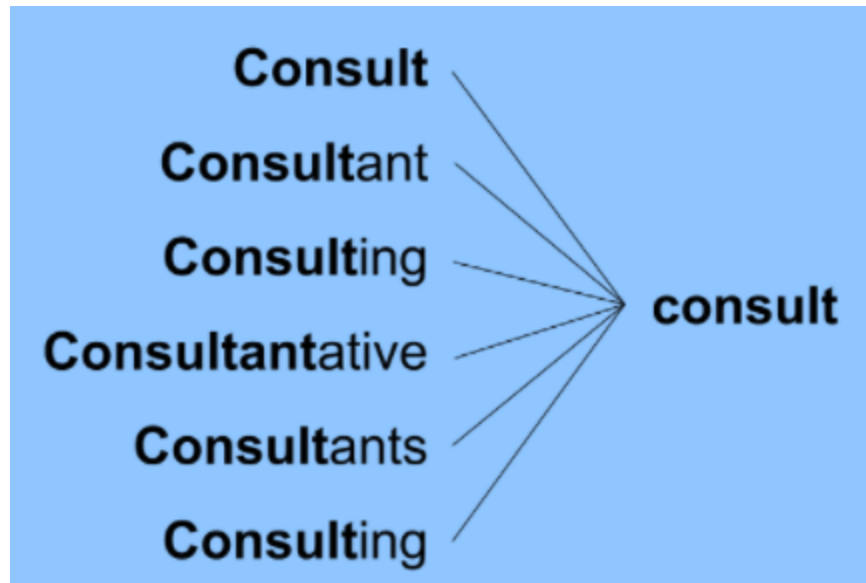
Special characters like ' , / * @ ' etc should be removed since they don't play any important role in any NLP operation.

- **Removing the stopwords**

Stopwords (words like of, for, him, her, what, from etc.) are another element of language that is not important for any basic NLP operation and removing them is a good practice.

- **Stemming/Lemmatization**

This is the standard step in NLP preprocessing where we convert words into their basic forms. This helps to get the right frequency of the similar words and saves us from treating similar words as different words. Here is an example:



3. Vectorization

This is the step where we convert the words into numbers. Since the computer doesn't understand text but only numbers, this is the step we need to take before applying any machine learning algorithm.

There are various techniques available to perform vectorization like CountVectorizer, TF-IDF and word-to-vec etc. We tried all three mentioned here and concluded that TF-IDF is the best technique for us to use.

TF-IDF vectorizer calculates the product of two terms of each word:

1. **Term Frequency** : Number of words a term t appears in a document d divided by total number of words in document d .
2. **Inverse Document Frequency** : Log of (Number of total documents/Number of documents containing term t)

TF-IDF does not consider each word as the same and gives those words more important that are not very common and also not very unique, which are the words that will help us to determine similarity between two documents.

After performing TF-IDF vectorization on the combination of all the columns, we got around 1800 terms in our dataset.

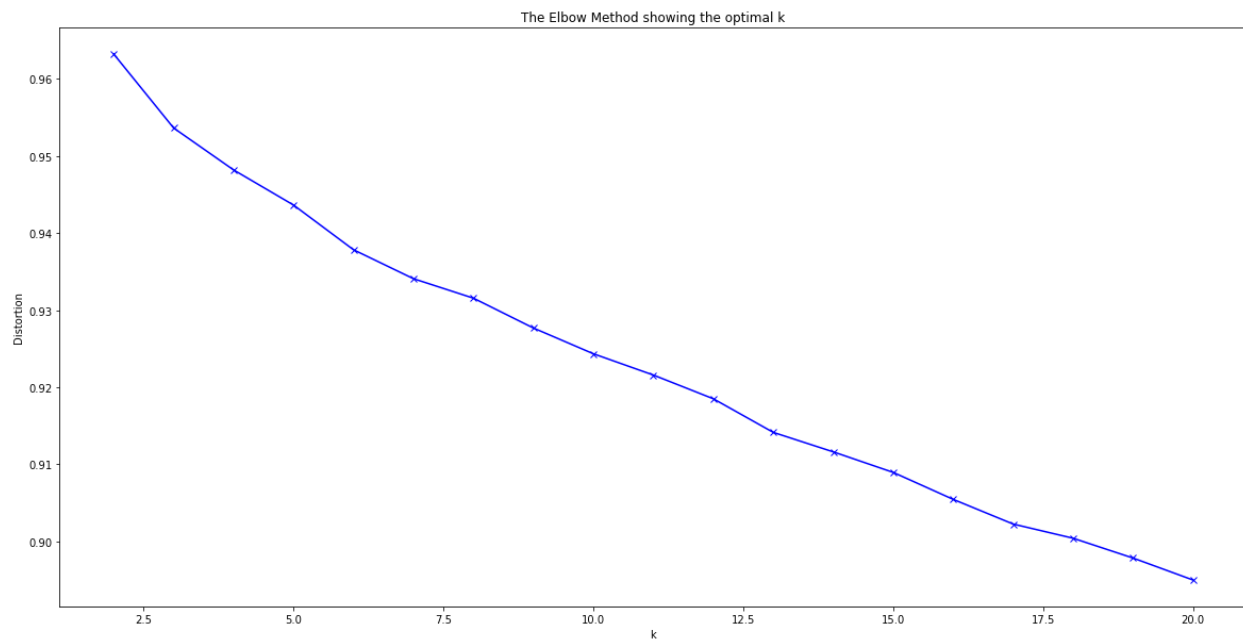
4. Clustering

This is the step where we actually apply some machine learning algorithms to classify each company in different groups. We had different options for what algorithm to apply like

K-Means and Hierarchical clustering, but finally we decided to go with K-Means clustering. Here are some points why we think K-Means clustering algorithm worked best for us:

- K-Means clustering algorithm is relatively less computationally expensive. Since our dataset is already huge, Hierarchical clustering will take forever to run.
- Deciding the number of clusters is also easier in K-Means since we have to run K-means a few times to decide it by the elbow curve.

Here is the graph of the WCSS (within cluster squared sum) score vs the number of clusters ranging between 2 to 20. We were not able to see a clear elbow but if we observe closely we can see a dent on $k=12$. So we decided to go ahead with 12 clusters.



5. Labelling the Clusters

Now since we have the different clusters from the last step, we have to name these clusters in this step. For giving a name to the clusters, we need to explore the text and topics on those clusters. For that we created word clouds on our clusters which are as below:

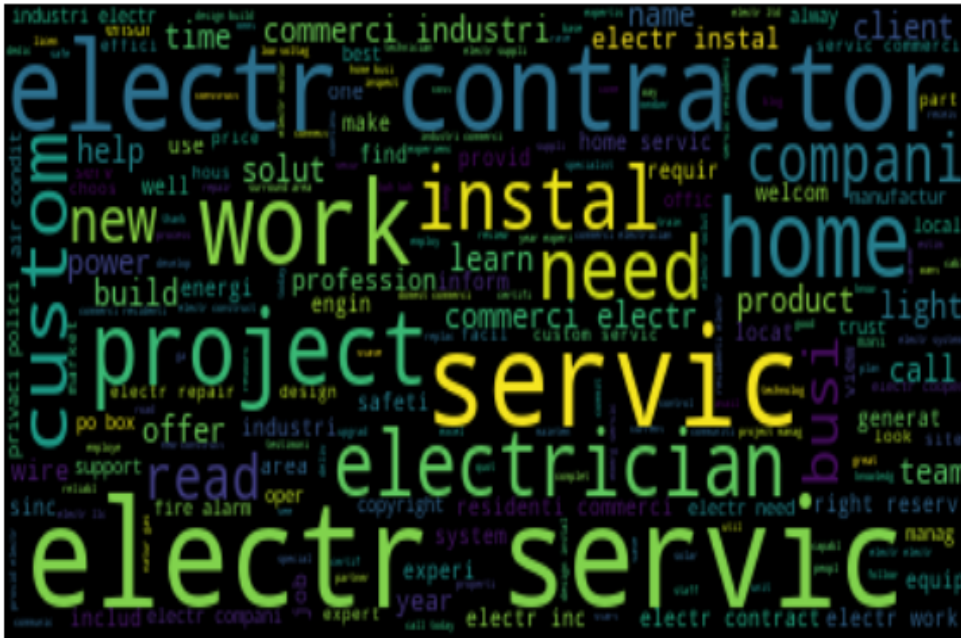
Pet Healthcare



E-commerce



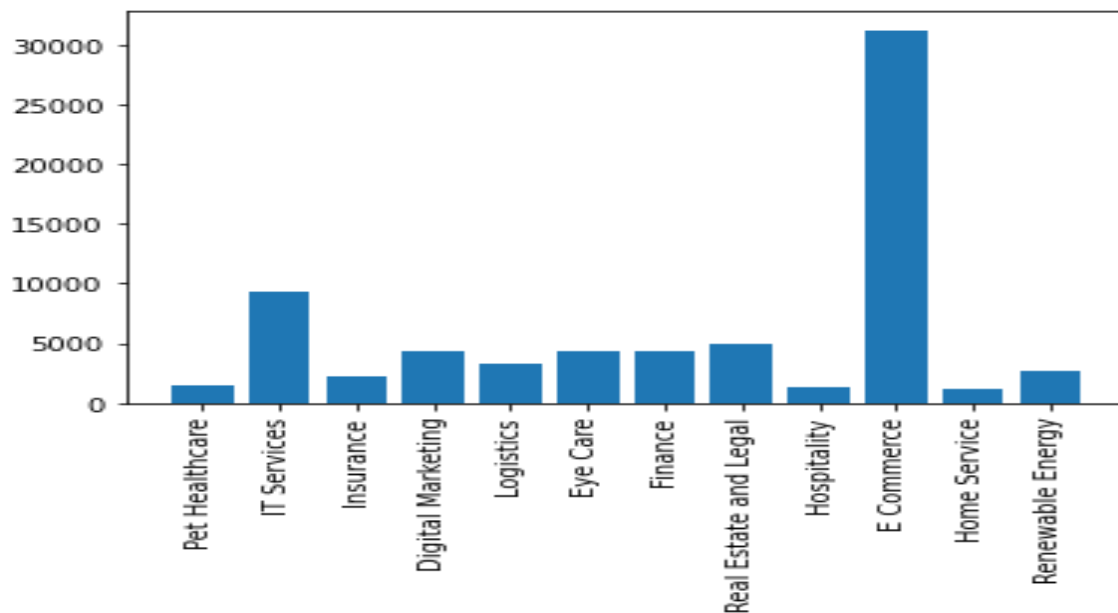
Home Services



Renewable energy resources



3. Distribution of the clusters on our dataset



Distribution of clusters on our dataset shows that a big number of companies in our dataset belong to e-commerce which includes all businesses and companies that provide services or products through the internet.

Challenges

We faced few challenges as discussed :

1. While text preprocessing we came across characters from different scripts and there were emojis in the text which was hard to find in the beginning.
2. There were a lot of null values we had to handle and those null values were unevenly distributed across the columns.
3. We had limited resources and the dataset was too huge and the algorithms were computationally expensive.
4. Deciding the number of optimal clusters was a bit difficult as the dataset was vast and we can get more clusters.

Conclusion

1. As we observed “E-commerce” was the top cluster from an overall perspective, we can further dig down and do sub-clustering to gain more insights.
2. We were able to categorize businesses/companies in 12 clusters based on different features and on the combination of whole features as well.
3. Those clusters represent various industries and we can approach those companies of each cluster with relevant business proposals.