Executive Summary of the Thesis

Natural Language Processing for startup industry classification

Laurea Magistrale in Computer Science and Engineering - Ingegneria Informatica

Author: Giuseppe Alessio Dimonte

Advisor: Prof. Licia Sbattella

Co-advisors: Vincenzo Scotti, Leyla Damoisaux-Delnoy

Academic year: 2021-2022

1. Introduction

The following work describes the activities related to the internship period carried out by the student at "Novable", a Belgian company providing AI-supported strategic consulting services and whose main product is a NLP-based startup scouting platform. In this perspective, the goal is the development of a functionality to perform startup industry classification starting from the text and the information present on each company website. The idea of performing industry classification enables innovation departments to track and monitor specific industries and to keep updated with specific trends, techs, and advancements in a fast, automatic and scalable way.
The proposed work outlines an evaluation and study of different methodologies for the creation of the functionality considering the most known methodologies and processing techniques in NLP.
The resulting model outputs a number of industries for each startup based on the text extracted in real-time from each website, allowing to filter and navigate the list of companies based on the industries and providing a different way of interaction within the company platform.

2. Background

Natural Language Processing refers to the intersection between Linguistics, Computer Science, Information Engineering and Artificial Intelligence. It deals with giving computers the ability of understanding text and spoken words in the most analogous way human beings can. The major advantages are of course the possibility of performing large-scale analysis and automating processes in real-time, requiring mostly no human intervention.
Nonetheless, if human beings can deal with raw text directly, Machine Learning algorithms cannot and for this reason it is important to preprocess text data and convert it into a set of features describing text and words in a numerical representation that the computer can understand and process. This phase is called "feature extraction" and the main methodologies are based on Bag of Words and Word Embeddings. BoW methodologies consider each document as a collection of words (vocabulary) and map each word with their related number of occurrences. In this way it is possible to compute the "Term Frequency", that gives more importance to more common words and the "Term Frequency - Inverse Document Frequency (TF-IDF)", which weights words according to their importance in

each document.
With Word Embeddings each word is represented as a n-dimensional vector in a multi-dimensional semantic space, so that the linguistic context of words can be reconstructed and the semantic relationships between words can be understood. Indeed, vectors closer to each other share common context and the vector representation enables to perform mathematical operations between different vectors in order to infer new information starting from already known vectors.
In the case of text classification keywords play a crucial role for determining the affiliation to a certain category. For this reason a specific focus is put in extracting the most important and descriptive words from the text. Certainly, TF-IDF is the method satisfying this requirement by weighting the relevance of specific words in relation to a specific document. The feature vector created using the TF-IDF vectorization is a sparse vector, whose size depends on the vocabulary considered and it rapidly increases when more words are taken into account. Nevertheless, sparser vectors do not guarantee better results, but instead contribute to the curse of dimensionality problem, causing an exponential increase in the computational efforts and more challenging predictive modeling tasks when the data dimensions increase.

3. Data

The first phase for every Data Science project consists in gathering the data and find the most useful set for performing the required operations. In this section the organization of the dataset and an analysis of the available data are discussed together with some solutions for tackling the balancing problem.

3.1. Collection

The primary source of data for Novable is a well-known data vendor providing business information related to private and public companies. The two entities have a contract, authorizing Novable to have access and to store vendor's data on a database by paying a fee each time data is refreshed. In this way it is possible to have access to a large amount of fresh data describing more than 1.7 million companies worldwide. In order to create a useful dataset for text classification it is important to map companies to their industry labels. After an analysis of the data, industry labels are spot in the column "tags" of the database. Labels are organized in 47 macro-categories, each one having multiple micro-categories for a total of 1019 micro-classes distributed among the set of macro-categories. For example, the macro-category "Artificial Intelligence" has "Machine Learning", "Natural Language Processing" and "Intelligent Systems" as micro-categories and similarly for all the other macro-categories.

3.2. Pre-processing and Encoding

Further analysis of the available data highlights that a company usually belongs to multiple industries at the same time. For example, a tech company like "Apple" is affiliated to “Computer”, “Consumer Electronics”, “Hardware” and "Software" industries concurrently. This means that not only these industries are topic-correlated, but also that a company cannot be mapped to one and only one label at a time. For this reason, the list of companies are organized in a directed graph data structure. In this way the connections between companies, labels and sub-industries are underlined and easily displayed. The graph-based data structure stores in each node the UUID and the industry labels for each specific company. Starting from the data structure is then possible to create functions to retrieve the required information and populate the dataset starting from the UUID.
The dataset is organized in different columns:

- **UUID**: 128-bit value uniquely identifying each company in the database
- **Text**: maximum ten pages are extracted and pre-processed from each company website, giving priority to the sections "About" and "Product"
- **Text vector**: 300-dimensional vector identifying each company in a condensed way by using a custom CBOW Word2Vec model on top of the set of keywords extracted from each website using a TF-IDF weighting technique
- **Label**: industry assigned to each company and retrieved from the data structure

3.3. Problems

Even thought companies falling in multiple industries at the same time are certainly a problem for classification tasks, it is also true that there is no unbiased answer for assessing to which macro-class the duplicated company belongs most and cleaning the duplicates following a predefined rule can lead to a biased model, compared to leave duplicated companies and let the model automatically deal with the presence of an element multiple times. The second problematic is related to the dataset unbalancing. For certain categories, like "Software" and "Financial Services", the number of samples is extremely disproportionate compared to other classes, as shown in Figure 1.

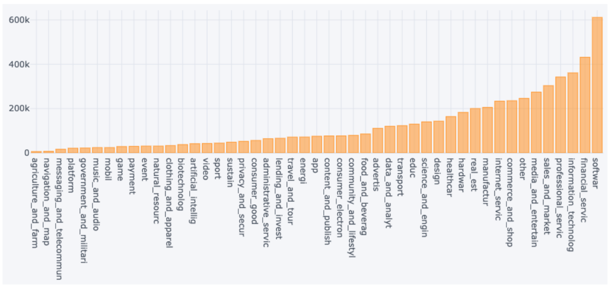

Figure 1: Macro-Classes Distribution.

In the literature various methodologies are suggested for trying to balancing the dataset. The first way-to-go is by down-sampling and reducing the number of the majority classes in order to balance the distribution. Nonetheless, in this way a large number of samples might be deleted together with useful information, but decreasing the amount of training samples helps reducing the training time and solving some memory problems. For the considered dataset the total number of samples goes from 1.7 million to 15 thousands, meaning that less than the 10% of the available dataset is used for training the model, which is clearly not a good way of using the available resources.
Secondly, in a similar way over-sampling can be used, by increasing the number of minority class members in the training set according to the number of samples in the majority class by duplicating the elements for the minority classes until balancing the dataset. The major advantage is that no information from the original training set is lost because all the samples are kept, but on the other hand it is prone to over-fitting since it is trained multiple times on the same data and the model ability of generalization is strongly affected.
Lastly, since up-sampling does not provide any additional information to the dataset, SMOTE (Synthetic Minority Oversampling Technique) can be used to synthesize new samples: starting from the minority classes' original data points, SMOTE algorithm performs data augmentation by creating new synthetic data points which are slightly different from the original ones.
However, none of these methodologies leads to better results compared to the original unbalanced version of the dataset, which is the one on top of which all the models presented in the next chapters are trained and tested.

4. Methodologies

This chapter describes how input text data is processed and it reports all the methodologies implemented for the task of startup industry classification, dividing the analysis in four main families: Statistical Learning, Ensemble Learning, Deep Learning and Language Models.

4.1. Text pre-processing and Vectorization

For Natural Language Processing text yields the information necessary to do predictions and extract insightful knowledge from it. Text pre-processing and vectorization steps might change depending on the desired task and they are not directly transferable from one assignment to another, but they may vary depending on the job required to achieve.
In the case under consideration the text coming from the websites follows a well-defined pipeline before being passed as input to the Machine Learning models: it is first lower-cased, then special characters, numbers and stop-words are removed, next the resulting words are stemmed and transformed in their base form and finally vectorized using the TF-IDF technique.

4.2. Models

Statistical Machine Learning, Feature Engineering, Artificial Neural Network, Attention-based models are implemented and tested, comparing their outcomes and fine-tuning their parameters to achieve the best overall performances.

4.2.1 Statistical Learning

The first methods used are models related to Statistical Machine Learning, dealing with the statistical inference problem of finding a predictive function based on the data, including the identification and interpretation of patterns and associations. According to the literature, some of the most known methodologies for text classification have been adapted to the task. These techniques are:

- Multinomial Logistic Regression (MLR)
- Multinomial Naïve Bayes (MNB)
- Linear Support Vector Machine (LSVM)
- Kernel Support Vector Machine (KSVM)

4.2.2 Ensemble Learning

Because of the unbalancing in the dataset also Ensemble techniques are tested. Indeed, they usually improve performances by generating and combining together multiple models, known as "base estimators". In fact, grouping together multiple models allows to solve particular learning tasks and several technical challenges, including high variance, low accuracy, features noise and bias. The tested techniques are:

- Random Forest (RF)
- XGBoost Classifier (XGBC)

4.2.3 Deep Learning

Next, Artificial Neural Networks are implemented. A Neural Network is a computational learning system using a network of functions to understand and translate a data input of one form into a desired output. The more examples and variety of inputs the program sees, the more accurate are the results. The tested architectures are:

- Multi-Layer Perceptron (MLP)
- LSTM

4.2.4 Language Model

Lastly, language models are evaluated. In NLP Language Models are probabilistic and statistical techniques used to determine the probability of a given sequence of words occurring in a sentence based on the previous words, and helping to predict which word is more likely to appear next in the sentence. Language models form the backbone of NLP, transforming qualitative information about text into quantitative information that machines can understand. They learn the features and characteristics of basic language and use those features and rules in language tasks to understand new phrases, to predict and produce new sentences.
During this phase BERT is used and fine-tuned for the task of classification. BERT is a machine learning model developed in 2018 by researchers at Google AI Language and it has been chosen because it presents state-of-the-art results in a wide variety of NLP tasks.

5. Evaluation and Results

This section provides a brief overview of the metrics used for assessing models' performances. Moreover, results are commented and an analysis related to the sparsity of the TF-IDF vector and the labels organization is done.

5.1. Evaluation approach

The metrics considered for comparing the models comprehend Accuracy, F1-score and an adhoc measure named "Adaptive Benchmarking (AB)". The proposed metric AB counts the number of True Positives among the total number of predictions generated for a specific entry. It is adaptive because the number of predictions in output depends on the probability value of each prediction, which is outputted only when it exceeds a certain threshold.
In order to train and test the models, the dataset is split in training set and testing set with a ratio of 9:1, allowing to have more samples for the training phase, as it is usually considered a good practice for training models on text data.
The statistical models are implemented using the Scikit-learn library, the neural networks are modeled using Tensorflow and Keras and BERT is coded using the PyTorch and the HuggingFace libraries. The fine-tuning phase consists in finding the best set of parameters to give as input to achieve the best overall performances by switching and trying new values, like the solver, the number of epochs to train and the maximum number of iteration to wait for convergence.

5.2. Results and Comments

In Table 1 it is possible to compare the evaluation metrics considering their behaviour on the text data vectorized with TF-IDF and based on the original unbalanced version of the dataset. This comparison highlights that MNB and RF models are bad both on Accuracy and F1-score, while all the other models are comparable. The best value for Accuracy is given by fine-tuning BERT and it is followed by KSVM, but none of them is able to generalize and achieve good performances when taking into consideration the AB metric, meaning that they are affected by the unbalancing in the dataset. Finally, the models with the highest AB metric are MLP, MLR and XGBC, which correctly predict 4 out of 5 labels on average.

	Accuracy	**F1-score**	**AB**
MLR	19%	18%	77%
MNB	14%	4%	-
LSVM	18%	17%	-
KSVM	21%	17%	39%
RF	13%	3%	-
XGBC	20%	18%	76%
MLP	20%	12%	82%
LSTM	20%	12%	55%
BERT	28%	4%	61%

Table 1: Results with Text

The TF-IDF vector is by definition a sparse vector with many values equal to 0 and some values equal to 1. For this reason its dimension can vary a lot depending on the size of the vocabulary. Hence, an analysis on the performances of the model depending on the maximum number of allowed features for the TF-IDF vectorizer is done. The plot shown in Figure 2 represents the performances of AB for MLR and MLP with a different number of features for the vectorizer. As demonstrated, for both the models the performances reach a good level after the first iterations and then slightly improve by adding features. This analysis is very insightful, proving that even with a smaller vocabulary the performances are quite stable and that training the model on a bigger vocabulary is not going to change the results drastically. In fact, a larger vocabulary means a bigger TF-IDF vector, and a bigger vector does not assure a significant improvement in terms of performances compared with the computational cost required. Moreover, in this way the problem of curse of dimensionality is solved, demonstrating that there is no need of dealing with huge sparse vectors to achieve good results and that it is possible to achieve the same degree of performances by using smaller vectors that require less computational power to be processed.

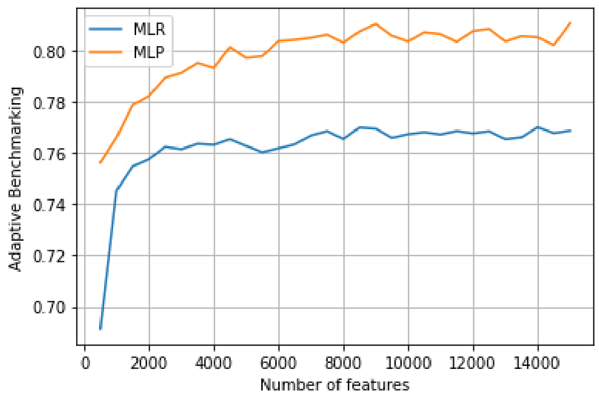

Figure 2: Relationship Number of Features and Adaptive Benchmarking.

In the analysis done so far 47 macro-categories were considered for testing various techniques. In order to reduce the dataset sparsity and try to improve the performances a meaningful approach consists in reducing the number of classes, deleting the categories which are not descriptive enough and merging together categories sharing the same context. Even though this approach might bias the dataset it was implemented with the help of business experts so that the risk of making mistakes is considerably reduced. Hence, seven categories are deleted and twenty-four are merged. Thanks to this semplification in the dataset the resulting categories are 24 and the performances for Multinomial Logistic Regression and Multi-Layer Perceptron increase by 10% and 5% respectively according to the Adaptive Benchmarking metric.

6. Productization

The final objective is the implementation of the functionality in a way that the company can benefit from, making it productizable and integrated with the current DeepMatching^{TM} algorithm and with Novable's web application.

Three ideas are proposed:

- Show the industry labels for the companies which do not have any assigned tag from the database: currently the unlabeled companies represent the 6% out of the 1.7 million companies on the database, but in the near future the company wants to become independent from the data vendor and autonomously assign the industry labels.
- Generate statistics and info-graphics for the marketing team, to gain visibility as a company and demonstrate the activities of the Data Science team.
- Use the industry classifier together with a UI to create a visual way for the user to filter out companies based on the industries it belongs to.

7. Conclusion

This part aims to conclude the analysis, providing an overview of the objectives of the work and wrapping up the methodologies analyzed and implemented. The conclusions and the findings of the work are clarified and some suggestions for future improvements are presented.

7.1. Concluding Remarks

The document provides an overview of the activity and research done for the development of the startup industry classification functionality using Natural Language Processing techniques. Throughout the dissertation the whole theories and procedures involved are explained and documented. In the first phase data sourcing, dataset creation and data analysis are performed before testing different classification techniques. Finally, considering all the models and according to the company needs regarding the way the functionality is intended to be used, the best solution proved to be Multi-Layer Perceptron, demonstrating that, even if more advanced techniques are currently in place, they should not be used as black boxes to solve any kind of problem. Additionally, thanks to the comparison between TF-IDF vectorization and Attention-based techniques it is possible to understand that the specific task of startup industry classification benefits of more satisfactory performances when taking into account the importance and occurrence of the words in the text rather than the context and the relationship between words.

7.2. Future Works

The work done and proposed has been tailored according to the company requirements and resources available. In line with the developments and the results achieved so far, some improvements are suggested for the future. The first one consists in exploring the classification in micro-classes and proposing a way to manage the high number of micro-categories, exploiting cascade and hierarchical classification techniques. Secondly, linked to Transformers and Attention-based techniques, a significant improvement for text classification might be achieved by taking into account different types of transformer architectures. A particular focus should be put on Transformers with the ability of computing longer sequences of tokens based on Sparse Attention Models, like Extended Transformer Construction (ETC) and BigBird.

References

[1] D.-J. Jurafksy, *Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 3rd ed. Stanford, 2022.

[2] V. Vapnik, *The Nature of Statistical Learning Theory (Information Science and Statistics)*, softcover reprint of hardcover 2nd ed. 2000 ed. Springer, 2010.

[3] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning, second edition (Adaptive Computation and Machine Learning series)*, second edition ed. The MIT Press, 2018.

[4] K. Kowsari, "Text Classification Algorithms: A Survey," 04 2019. [Online]. Available: https://arxiv.org/abs/1904.08067