Classifying Reurets-7083 dataset. A Multinominal Naïve Bayes approach

Author: Ioana-Maria Popescu

Abstract

The area of artificial intelligence that has drawn the most interest from researchers in recent years is natural language processing. In order to perform research and build natural language processing (NLP) and machine learning systems, Reuters offers a sizable corpus of articles. The previous methods that have consistently produced the greatest results range from applying word embedding regularization together with a vector space model representation to more sophisticated message passing attention networks that can comprehend and categorize texts from this Corpus. The objective of the current study is to construct a system that can categorize the documents in the Reuters-7083 dataset utilizing a few well-known pre-processing techniques and a Multinomial Naive Bayes algorithm.

1 Introduction

Data is to this century what oil was to the previous one, driven by modern technology advancements. The collection and distribution of vast amounts of data today parachutes our world. With so much material being shared online, machine learning algorithms must be created that can automatically condense lengthy texts into concise summaries and provide accurate summaries that elegantly convey the intended messages. [1]

The primary goal of this study was to offer a novel method for text mining and document categorization by going through several pre-processing stages and illuminating each of the outcomes. Similar studies have been published during the last three years, with some of them producing unexpectedly excellent findings. According to Vit Novotny et recent study, employing word embedding regularization in conjunction with a vector space model representation yields a prediction accuracy of 92.65 on the Reuters Corpus. The outcomes provide a fresh perspective on how to create a text classification system.

The essay is organized as follows for the remaining sections: We describe both the feature selection procedure and the whole pre-processing stage in Section 2. The representation of our data and the use of various Python libraries are covered in Section 3. The paper will be concluded in the last section, Section 4.

2 Pre-processing and Feature Selection

The first step of the implementation is to specify a set of rules which all the words need to follow to have a unified and coherent set of words. These rules include eliminating punctuation, numbers, and white spaces. Once this step is over, all the words and topics are going to be extracted

from documents. To do that, the implemented project search key phrases presented in all given documents like in a pattern and extract them accordingly. As an example, a list of words is created by fetching all the words contained in HTML tags such as "<text>" and "</text>", respectively "<title>" and "</title>". The implementation for getting the document's topics is like the one explained above, but the searched tag is "metadata/codes". Simulatively, the words and topics are counted to have an idea of the most and least important words and topics in the documents.

In the next pre-processing step, the stopwords need to be eliminated and convert all characters to lower case and remove numbers and words that contain numbers. The words with special characters, such as „@" which is typically used to describe an email address, will also be eliminated, along with the punctuation marks. We will extract the stem of each word in order to avoid utilizing numerous variations of the same phrase and instead keep the stem because the majority of our words now reflect real characteristics that can be used later.

Once the previous step is done and the words are ready to be processed, the most useful documents come into play. We eliminated the once's who don't keep the important information. We also excluded the topics/classes that are the most and the least prominent in documents, eliminating done with the 5-95% method based on multiple simulations results, which are showed in Table 1.

Table 1. Classes filtering methods.

Method	*Dataset documents*	*Number of classes*	*Number of words*
No filter	7083	68	23247
5-95 rule	6826	15	22432
10-90 rule	4294	5	12955

By filtering and eliminating classes that have the most amount (95%) and the least amount (5%) appearance, not only we reduced the number of documents and classes, but we also reduced the number of words.

Now that we've managed to erase a few classes along with the remaining unclassified documents, the next step would be figuring out how to identify which class is ideal for each document. Currently, each of the remaining documents fits into one or more classes. Working with lists of subjects that are varied lengths is irrelevant, so we will develop a method that gives each document a unique class. The method described by our categorization system will be applied to each of the 15 remaining classes, which will then be ranked according to how frequently they occur. By the end, each document will fall under no more than one class, the predominant class.

The next step is to determine how to reduce the number of words. This stage involves selecting features. Information gain is the method we chose to decide which qualities are valuable enough to keep or not. In order to quantify the information gain, we first calculated the entropy of each data set.

Both of the next definitions are taken from professor Daniel I. Morariu [2].

The definition of the entropy is:

$$Entropy(D) = \sum_{i=1}^{c} p_i log_2(p_i) \quad (1)$$

where D is a collection of n documents grouped into c classes and pi is the proportion of elements of D belonging to class i.

After defining entropy, we'll move on to describe the operation of our feature selection process, so in the next step, we calculate the information gate having the next definition:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \quad (2)$$

where Values(A) is the set of all possible values for the word A, and Sv is the subset of S for which A is equal to v.

We choose to label each attribute as follows in order to be able to retrieve useful data using the Information Gain method calculated before: 0 – if a word is not present in a document, 1 – if a word appears below average in a document and 2 – if a word appears above average. The average of a word indicates the number of occurrences of said word across the entire dataset.

3 Dataset representation

After outlining the full pre-processing steps, we underwent, we will now go through how our dataset is represented and how we will fit it to the NB technique. As we've just seen above, the data needed to be organized in order for the Information Gain technique to utilise it. We will return to dictionaries with key-value pairs, where the value indicates how frequently each attribute appears in a specific document. Labelling each attribute according to its frequency does not enhance the learning algorithm.

The Information Gain method was used to filter our global dictionary so that it now only contains the most crucial features. The next step is to update each local dictionary, removing any superfluous characteristics and keeping only those that are listed in the global dictionary. Upon updating each local dictionary, the entire dataset representation will be finished.

	usa	unveil	intranet	product	tuesday	intranetw	softwar	intend	establish
2504NEWS.XML	1	1	2	2	1	2	2	1	1
2538NEWS.XML	1	1	0	1	1	0	1	0	0
2775NEWS.XML	1	0	0	1	1	0	1	0	0
2792NEWS.XML	1	0	0	1	1	0	2	0	0
2822NEWS.XML	1	0	0	0	1	0	2	0	0
2836NEWS.XML	1	0	1	0	1	0	1	0	0
2848NEWS.XML	1	0	0	0	1	0	0	0	0
2917NEWS.XML	1	0	2	2	0	2	1	0	0
2955NEWS.XML	1	0	0	1	1	0	1	0	0
2978NEWS.XML	1	0	1	2	0	0	2	0	0
2982NEWS.XML	1	0	0	0	0	0	0	0	0
2984NEWS.XML	1	0	0	0	0	0	1	0	0
2988NEWS.XML	1	0	0	0	0	0	1	0	0
3665NEWS.XML	1	0	0	0	1	0	0	0	1
3785NEWS.XML	1	1	2	1	1	2	2	0	1
3813NEWS.XML	1	0	0	1	1	0	0	0	0

Figure 1. Normalized scores.

Figure 1 displays a representation of a proportion of dataset together with the frequency of words used in each document. The information displayed is meant to be kept in our multidimensional array, where each column index represents a word and each row index represents a document.

As we discussed in the first portion of this paper, our strategy depends on creating a text mining system using the Python programming language, so in order to fit our data to the algorithm, we transformed it into the special Python array type NumPy in order to be able to fit it to the training method.

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. [3]

1	1	2	2
1	1	0	1
1	0	0	1
1	0	0	1
1	0	0	0
1	0	1	0
1	0	0	0
1	0	2	2

Figure 2. Dataset represented as a matrix.

Our final dataset is shown as a NumPy matrix in Figure 2. In order to retain our data in a data structure that can be used to train the NB method, we can see by comparing the matrix with the previous figure, Figure 1, that each attribute occurrence is taken from each page.

4 Conclusions

In this study, the performance of the Multinomial Naive Bayes algorithm is examined together with the primary pre-processing stages of developing a document classification system, Information Gain feature selection method, and feature selection. Many Python open source modules were used in our tests, and they were very helpful for our research from the perspectives of both time efficiency and algorithm complexity.

5 References

Acknowledgement: This work was supervised by Professor Daniel I. Morariu, from „Lucian Blaga" University of Sibiu, Engineering Faculty, Computer Science and Electrical and Electronics Engineering Department.

[1] The PyCoach, Is Data The New Oil of the 21st Century or Just an Overrated Asset?, https://towardsdatascience.com/is-data-the-new-oil-of-the-21st-century-or-just-an-overrated-asset-1dbb05b8ccdf, 2022
[2] Daniel I. Morariu, Feature Selection – Entropy and Information Gain, Classroom, 2022
[3] Wikipedia, NumPy, https://en.wikipedia.org/wiki/NumPy, 2023

Ioana-Maria Popescu
"Lucian Blaga" University of Sibiu
Advanced Computing Systems
Computer Science and Electrical and Electronics Engineering Department
Romania
E-mail: maria.popescu@ulbsibiu.ro