Building a simple binary classifier on text data

Ewelina Gajewska

1 Abstract

The aim of this project was to build a simple binary classifier on text data. I chose to classify texts by sports disciplines they belong to - either football or boxing. Overall, I collected 609 instances of text from six sports websites, then cleaned the data using spaCy and train my two Naive Bayes models (which vary in the type of applied vectorizer) using scikit-learn. Both models' results were very high - all scores greater than or equal to 98%.

2 Introduction

Machine learning methods are among fundamental methods used in the field of natural language processing (Aggarwal, Zhai, 2012). They are used, for example, in web search engines for information retrieval (Aggarwal, Zhai, 2012; Bird, Klein, Loper, 2009). Thereby, a user looking for specific information gets results relevant to the searched topic. I wanted my model to learn to classify sports news into separate categories, similarly to automated methods available in search engines (e.g. Bookmark Folder app). In the light of available algorithms I followed scikit-learn documentation (Pedregosa et al., 2011) in selecting the right estimator. Therefore, I chose Multinomial Naive Bayes classifier. This is one of the simplest and most popular models in the field of supervised machine learning (Aggarwal, Zhai, 2012). Naive Bayes is also among the most efficient and effective classifiers (Grimmer, Stewart,

2013). It chooses the right label for a given text instance based on probabilities (the highest value wins). In other words, for every text sample a label probability is estimated through multiplying a probability of a specific label by a sum of probabilities of all terms in the sample given the label (Bird, Klein, Loper, 2009).

The project required text data that differs in some respect and thus can be classified into one of two categories. I chose to train my model to classify a given piece of text into either the "football" category or to the "boxing" category. They both fall into a broader sport category. At the same time, however, they are very distinct and specific disciplines, also in the terms of popular phrases they employ. A similar work can be found in (Grimmer, Stewart, 2013) where authors analyzed and compared supervised and unsupervised machine learning methods on examples from politics, military and health. In addition to the classification task, I decided to compare two types of vectorizers, i.e. Count Vectorizer and Tfidf Vectorizer, in order to verify which one is better for this kind of simple classification. The former computes words frequencies in every text instance given the language corpus. The latter encodes for each term a normalized and weighted vector regarding its frequency in the whole corpus. A similar approach was taken by Basarkar (2017) where learning model consisting of TfIdf Vectorizer and Naive Bayes classifier performed slightly better than the one with Count Vectorizer in general (4% difference in accuracy) as well as in two sports classes.

Link to my Google Colaboratory project file: `https://colab.research.google.com/drive/1NCvUX3lkjdpZHCSfwTtF65WQfHmYWS4e?usp=sharing`

3 Methodology

Following Grimmer and Stewart's (2013) three step process, in this project I distinguished three stages: 1) collecting data, 2) preprocessing (cleaning) data and 3) building and validating a model. The first thing was choosing a

type of and collecting text data. I decided to use texts from online websites belonging to two categories of sports. I gathered data through web scraping using BeautifulSoup and Requests Python libraries as well as manually by hand. Overall I collected 609 instances of text, 304 in "football" category and 305 in "boxing" category. It was derived from sports news headlines and articles from the following websites: ESPN, BBC Sport, Yahoo! Sports, Sky Sports, CBS Sports and The Athletic. The second step in the project was to preprocess (clean) the data. Thus, I assigned label "1" to texts from the "football" category and label "0" to texts from the "boxing" category. Then I transformed all characters into lowercase as well as removed stop words (i.e., words that do not convey much meaning), punctuation and numbers using spaCy library. Also, all words were tokenized and lemmatized. I did not used stemming because it cuts off meaningful parts of words leaving only root forms (stems). Using lemmatization I could get words in their dictionary form and at the same time maintain context associated with them. The last stage was splitting the data into train and test sets (the ratio was 0.8 and 0.2, respectively) and assigning a random state number for the reproducibility of results. Finally, the last thing was to vectorize text data, train models and evaluate them.

Model name	Class label	Precision	Recall	F1-score	Model accuracy
TF-IDF Vectorizer and Multinomial Naive Bayes	0	0.98	0.98	0.98	0.98
	1	0.98	0.98	0.98	
Count Vectorizer and Multinomial Naive Bayes	0	0.99	1.00	0.99	0.99
	1	1.00	0.98	0.99	

Table 1: Models' confusion matrix.

4 Results

Detailed models' performance metrics can be found in Table 1. Accuracy of the model with Count Vectorizer was 0.992 with only one false negative

Figure 1: Football category most common 10 words.

and therefore performed slightly better than the model with Tfidf Vectorizer -0.984. This difference results from one additional prediction error (false positive). Notwithstanding, both models' scores were very high. Also, I plotted two subsets of processed texts using WorldCloud to discover ten most common words in each class. The plots can be found in Figure 1 for the "football" class and in Figure 2 for the "boxing" class.

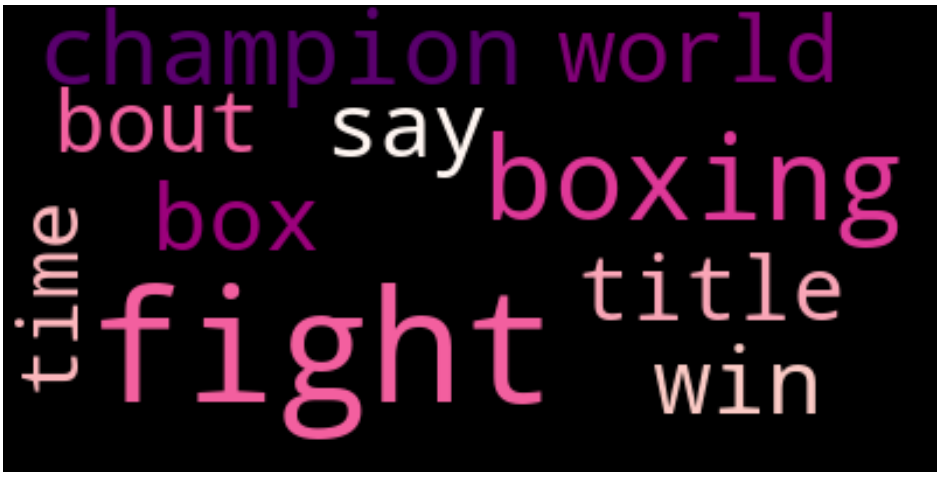

Figure 2: Boxing category most common 10 words.

5 Conclusion

Supervised learning models built in this project performed surprisingly well, their scores were very high. Even though classification task was simple, chosen categories were quite similar because both of them are sports disciplines and therefore they share a common vocabulary (for example, word "win"). In future work there could be implemented multi-label classification model using similar sports categories in order to verify these results in a more complex task.

References

Aggarwal, C. C., Zhai, C. (Eds.). (2012). *Mining text data.* Springer Science Business Media.

Basarkar, A. (2017). Document Classification using Machine Learning.

Bird, S., Klein, E., Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit.* O'Reilly Media, Inc.

Grimmer, J., Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political analysis, 21*(3), 267-297.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research, 12*, 2825-2830.

`https://scikit-learn.org/stable/`