International Journal
Information Technology Applications (ITA)
Volume 7, Number 1, July 2018

TEXT CATEGORISATION USING MACHINE LEARNING AND NAÏVE BAYES CLASSIFIER

Marek Špilda

Abstract:

The aim of this paper is to explore, implement and verify the functionality of Naïve Bayes classifier to classify natural language texts into categories. A sample classifier was developed to evaluate the results. For text cleaning, different text features were labelled and tests were performed with different combinations of cleaning methods to evaluate its effects on the categorization accuracy.

Keywords:

machine learning, text classification, Naive Bayes.

Introduction

With the rise of electronic media and the Internet, the number of email and chat interactions between companies and their customers has largely increased. Companies need to distribute a growing number of written customer requests to specific departments and their qualified employees. Categorization algorithms will make this process more efficient in distribution of inquiries to the right personnel.

Automatic classification can also be used in processing other various texts such as newspaper and magazine articles, detecting spam messages and various fraud schemas, categorize social media posts and many other types of written texts.

This paper aims to describe different types of machine learning methods and explain, how Naïve Bayes algorithm works and show experimental results of classification of movie reviews into two different classes (1 star and 10 star reviews) with different types of text cleaning methods including text-only, lower and mixed case text or using special labels for question and exclamation marks, numbers or emoticons.

1 Machine learning

Machine learning is a subset of artificial intelligence field, that contains algorithms able to learn without having to implement strict processing of data sets. Machine learning explores the possibilities to process large amounts of data, find hidden patterns between data features and use the learned knowledge to work with new and unknown data. The main machine learning methods are:

- supervised learning
- unsupervised learning
- semi-supervised learning
- reinforcement learning

Supervised learning uses examples with existing inputs and outputs to train and then predict the outcomes of new events.

Unsupervised learning may be used, when the outputs, categories or labels, are not known in advance. The algorithms look for hidden relationships and patterns in given data.

Semi-supervised learning is a combination between supervised and unsupervised learning. Usually the input consists of subset of data including related output results (categories or labels) as well as subset of data without its outputs.

Reinforcement learning requires feedback, using which the algorithm learns on the rightness of its outcomes and gradually iterates to more accurate results.

2 Naïve Bayes classifier

Naive Bayes classifier represents a statistical classifier and uses the supervised learning method. Classifier predicts the class by calculating each class probability based on the Bayes' conditional probability. Conditional probability is calculated using following equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

A and B are some events. $P(A|B)$ is a probability of occurrence of event A, if B is valid. $P(B|A)$ is a probability of event B, if A is valid. $P(A)$ and $P(B)$ are probabilities of events A and B respectively. With the Naïve Bayes classifier, there will be k classes $C_1, C_2, \dots, C_k$. Each tested document will be represented as a set of words, $X = \{x_1, x_2, \dots, x_n\}$, where x_i are single words in the tested document. The classifier will predict, into which class C the document X belongs by calculating the probability for each class. The result will be the class with the highest probability.

Naïve Bayes uses the concept of so called "bag of words". This means the order of the words will not matter. Example givne, for the algorithm, a document "This is a good movie" is exactly the same as "This movie good is a". That is also the origin of the name "Naïve" – the probabilities are "naively" multiplied without considering the order of the words themselves.

To calculate the class, following equation is used, where c_{nb} is the predicted class and w_i is the word at position i. The positions are indexes of words in the tested document.

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i \in positions} P(w_i|c) \tag{2}$$

An issue may arise if a tested document contains words, which are not in the training set. This would mean working with zero probability, giving a zero final result. To overcome this issue Laplace smoothing is introduced, where the number of occurrences of each word is increased by 1.

With documents containing significant amount of words, many very small fractions will be multiplied, resulting in floating point underflow. Hence the calculation will be done in log space. The final equations will be as follows:

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) + \sum_{i \in positions} logP(w_i|c) \tag{3}$$

3. Sample calculation

To demonstrate how Naïve Bayes classification algorithm works, a simple experiment will be used. The goal of this experiment is to utilize Naive Bayes to classify movie reviews into two different classes (1 star and 10 star reviews).

The following dataset contains a total number of 5 simple documents: 3 belonging to the 10 star class and 2 to 1 star class.:

Class C_{10star} - 10 star class documents:

1. the movie was good.
2. I liked the movie, great actors
3. great movie, star actors

Class C_{1star} - 1 star class documents:

1. bad movie
2. bad performance, bad actors

First, an unordered set of all distinct words present in the training set is created – the bag of words defined in the previous part of this paper. From a data mining perspective, for each document in the training set a total number of occurences in the bag of word set is determined.

The bag of words representation of each of the classes is described in the next tables.

Table 1 – Bag of words representation of documents in class C_{10star}

#DOC	the	movie	was	good	I	liked	great	actors	star	bad	performance
1	1	1	1	1	0	0	0	0	0	0	0
2	1	1	0	0	1	1	1	1	0	0	0
3	0	1	0	0	0	0	1	1	1	0	0

Table 2 – Bag of words representation of documents in class C_{1star}

#DOC	the	movie	was	good	I	liked	great	actors	star	bad	performance
1	0	1	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	1	0	2	1

$$P(C_{10star}) = \frac{3}{5} \tag{4}$$

$$P(C_{1star}) = \frac{2}{5} \tag{5}$$

$$P(w_i\,|c) = \frac{n_i+1}{n+vocabulary} \tag{6}$$

where vocabulary represents the total number of distinct words in the bag of words and n represents the total number of words present in the class.

A sample document “very good movie” would yield following results for the two classes:

$$P(C_{10star}) = \frac{3}{5} * P(very|C_{10star}) * P(good|C_{10star}) * P(movie|C_{10star}) =$$
$$= \frac{3}{5} * \left[\frac{0+1}{14+11}\right] * \left[\frac{1+1}{14+11}\right] * \left[\frac{3+1}{14+11}\right] =$$
$$= 0.6 * 0.04 * 0.08 * 0.16 = \mathbf{0.0003072} \tag{7}$$

$$P(C_{1star}) = \frac{2}{5} * P(very|C_{1star}) * P(good|C_{1star}) * P(movie|C_{1star}) =$$
$$= \frac{2}{5} * \left[\frac{0+1}{6+11}\right] * \left[\frac{0+1}{6+11}\right] * \left[\frac{1+1}{6+11}\right] =$$
$$= 0.4 * 0.059 * 0.059 * 0.118 = \mathbf{0.0001643} \tag{8}$$

$$P(C_{10star}) > P(C_{1star}) \quad (9)$$

The calculated values of conditional probabilities are consequently compared and classified as belonging to the group with higher probality. Therefore the sample document „very good movie“ is labeled class C_{10star}.

4 Implementation and evaluation

Although there are many Naïve Bayes classifier libraries available, own implementation of classifier was programmed. The sample classifier for this project was programmed using Python programming language. Python was chosen for its simplicity in creating quick prototypes, ease of use and a steep learning curve.

Program creates class objects which hold the word strings (the bag of words) and related information for each of the classes. A separate instance of the class is also used for testing data, as the tested data are also treated as bag of words.

In order to be able to easily turn on or off each of the data cleaning methods, a strings utility was implemented. Using this, the program allows to test all possible combinations by looping through all the on/off switch combinations. The results for the whole run are then stored in one final table.

For result processing, python library Pandas was used. Pandas is a library to hold large data frames and allows fast data processing. In addition, one line csv and excel exports are within the available methods, which help to reduce the burden of non-project core related tasks.

The data set used was extracted from user reviews of randomly selected movies from the IMDB database. An experimental set containing 2999 1-star and 2999 10-star movie reviews was created. 2000 reviews from each class were used for training and 999 reviews from each class were used for testing.

In addition, several different data cleaning methods were used during data Preparation phase and all of their combinations were tested. Methods for converting to lowercase, extracting numbers to be treated separately, extracting positive and negative emoticons, extracting question and exclamation marks and also for removing all non-alphanumeric characters were implemented.

Another method to clean data is to remove stop-words. Stop words are words with no special significance to the information carried by the test. In English texts, words such as ‘a’, ‘the’, ‘it’, ‘is’, etc. are all considered to be stop-words. A stop-word removal was also used in some tests.

In the results table, only results subset with the most significant cleaning methods is shown.

Table 3 - result subset with cleaned testing data

Success	TotalTests	Total Match	tolower Case	only Letters	remove Stop Words
87,64%	1998	1751	FALSE	FALSE	FALSE
87,64%	1998	1751	TRUE	FALSE	FALSE
87,19%	1998	1742	TRUE	FALSE	TRUE
86,49%	1998	1728	FALSE	FALSE	TRUE
86,44%	1998	1727	FALSE	TRUE	FALSE
86,14%	1998	1721	TRUE	TRUE	FALSE
85,69%	1998	1712	TRUE	TRUE	TRUE
85,64%	1998	1711	FALSE	TRUE	TRUE

Interesting discovery was, that the algorithm showed event better performance, when only the training set was cleaned, while leaving the testing set untouched. The results of successful matches were up to 90%.

Table 4 - result subset without cleaned testing data

Success	Total Tests	Total Match	tolower Case	only Letters	remove Stop Words
90,49%	1998	1808	TRUE	FALSE	FALSE
89,84%	1998	1795	TRUE	FALSE	TRUE
88,29%	1998	1764	TRUE	TRUE	FALSE
87,64%	1998	1751	FALSE	FALSE	FALSE
86,54%	1998	1729	TRUE	TRUE	TRUE
86,09%	1998	1720	FALSE	FALSE	TRUE
85,79%	1998	1714	FALSE	TRUE	FALSE
82,48%	1998	1648	FALSE	TRUE	TRUE

As it turned out, the most significant cleaning method was conversion to lowercase. Leaving out non-alphanumeric characters had small impact, but not positive. Least significant cleaning methods were marking emoticons and marking question and exclamation marks and marking numbers. These are not presented in the results table, as the differences varied in no more than fractions of percentage.

The stop words removal was expected to slightly improve the process. Measurements show, this assumption is not true. Removing stop words in all cases resulted in slightly worse classification outcome.

In general, Naïve Bayes gives good results for classification. The algorithm is not difficult to implement and does not require significant processing power for classification. It shows to be a good and cheap method to be considered for natural language document classification tasks.

Conclusion

Even though the algorithm is relatively simple to implement, the results are satisfying. Since the effect of the labelling emoticons, question and exclamation marks and marking numbers, was insignificant, in commercial applications this would be of no meaning. Naïve Bayes should be used as its name suggests – naively calculate the probabilities without any extensive data optimization.

Natural language processing is expected extend the possibilities, how people interact with technology. In addition, it provides many opportunities for exploration and scientific as well as commercial experiments.

References

[1] Jurafsky, D., Martin, J. H. (Draft of August 28, 2017) Speech and Language Processing, Third edition draft, https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf

[2] Busemann, S., Schmeier, S., Arens, R. G. Message Classification in the Call Center, DFKI GmbH, Germany

[3] www.python.org/doc

[4] Barber, D. (DRAFT November 6, 2012) Bayesian Reasoning and Machine Learning, Cambridge University Press, http://www.cs.ucl.ac.uk/staff/d.barber/brml/

Bc. Marek Špilda
Faculty of Informatics, Pan-European University in Bratislava, Slovakia
marek.spilda@gmail.com