iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 7 **Issue:** III **Month of publication:** March 2019

DOI: http://doi.org/10.22214/ijraset.2019.3437

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 7 Issue III, Mar 2019- Available at www.ijraset.com

Improving the Accuracy of the Naïve Bayes Classifier Using Feature Selection

Swapnil Arya[1], Prachi Nagane[2], Manish Rana[3]
[1,2,3]*Computer Department, Mumbai University*

Abstract— Machine learning history has a long list of problems regarding classification. One of the simplest and well performing classifier is Naïve Bayes Model. However, a natural issue that occurs with this classifier is that all attributes used to explain an instance are assumed to be conditionally independent given the class. This inevitably results is a decrement in the accuracy of the classifier and also due to interaction omission. In this paper, we focus to put more importance on more accurate model using feature selection technique which focuses on eliminating low information feature. Chi square method is explained which boosts the process of feature selection thereby improving efficiency of the system.
Keywords— ***Naïve Bayes, Multinomial, Sentiment, Classification, Feature Selection, Chi Square.***

I. INTRODUCTION

Within different types of machine learning issues, the most common attention is to solve flat classification problems. In the classification problem, the algorithm is given a set of labelled training set, each of them defined by a set of features, and the aim is to estimate the label of unknown labelled objects based on features. In a flat classification problem, every test example e (unseen during training) will get assigned a class c ∈ C (where C is the set of classes of the given problem), where C has a "flat" structure (there is no relationship among the classes). This approach is often single label, i.e. the classifier will only output one possible class for each test example. Apart from flat classification, there occur problems wherein features are dependent are problems that are hierarchical by its nature, consisting a hierarchy of classes to be predicted. With larger number of topics, flat categorizers face problems regarding complexity that may result in rapid increase of storage and time. For this reason, using flat classification algorithms might not be fit to the problem. The machine learning method has to be taught to deal with the hierarchical class structure. In this paper, we extend the traditional Naive Bayes to deal with a hierarchical classification problem. This paper explains how multinomial distribution helps to achieve hierarchical nature of text classification called Sentiment Analysis; also, how appropriate data techniques helps to achieve greater efficiency by categorizing text based on positive or negative polarities. Extracting sentiment from a body of text to determine the writer's attitude is generally known as sentiment analysis or opinion mining. The multinomial term shows that our features carry out a multinomial distribution, which says that we have multiple features and we count occurrences of feature or the relative frequency of each feature. For text classification, where words are considered features, a multinomial Naïve Bayes classifier distinguishes a document considering the relative frequency or the count with which a word appears in a document [1]. We intend to use proposed model on a considered dataset by showcasing its efficiency compared to traditional Naïve Bayes model.

II. WORKING OF A STANDARD NAÏVE BAYES CLASSIFIER

Naive Bayes classifiers are a set of classification techniques based on Bayes' Theorem. Naive Bayes is frequently utilized as a gauge in content characterization since it is simple and quick to execute. Naive Bayes has been regarded as "the punching bag of classifiers" [2] and has earned the doubtful distinction of placing last or near last in numerous head-to-head classification papers. [3] Still, it is frequently used for text classification because it is very easy and fast to execute. There are three different types of Naïve Bayes model based on the distribution techniques used: Gaussian, Multinomial and Bernoulli [4]. A member of Naïve Bayes Classifier family is a Multinomial Naïve Bayes Classifier. The term "multinomial" shows that our text or features follow multinomial distribution, which indicates that there are lot of features and algorithm that counts relative frequency of each feature. Sentiment Analysis prefers Binary Multinomial which concludes results in 1 or 0 [4]. Multinomial classifier works as follows:

Let,

D – Document

c – Class; for example, positive and negative are the two classes in sentiment analysis.

Text is often treated as bag of words where exact order of words is not a concern. Document characterization is done by the words present in it. Each word is treated as a feature and a document as a set of features f1, f2, f3 etc. A document contains

features and documents define classes, thus each class is defined by features. A Naïve assumption is made that each feature is independent of the other.

Pr(f1, f2, f3,..| c) = Pr(f1 | c). Pr(f1 | c) Pr(f2 | c) Pr(f3 | c)…. Pr(fn | c)

To predict a class for a document, first requirement is to find ĉ, max of {Pr(f1 | c). Pr(f1 | c) Pr(f2 | c) Pr(f3 | c)…. Pr(fn | c)}.

$$\hat{c} = \max_{c \in C} \Pr(c) \prod_{i=1}^{n} \Pr(w_i \mid c)$$

[4]

This can be considered as a new point (Dnew, ĉ). This says that, for a given observation, calculate the probabilities each class may have generated the sample and return the class that gave the highest probability. This is a class whose features resemble Dnew best and therefore is most likely to have generated the observation.

We use a set of documents labelled (d1, c1),..., (dn, cn) by classes to train our model. From the training dataset, we find P(c), the prior probability, and P(f1|c)•P(f2|c)• ... •P(fn|c), the likelihood probability for each class and values are stored in some data structure. P(ci) is the probability a training set document is in class ci.

$$\begin{aligned} \Pr(c_i) &= \frac{\text{number of docs of class } c}{\text{total number of docs in training dataset}} \\ &= \frac{N_c}{N_{docs}} \end{aligned}$$

[4]

P(wi|ci) is the fraction of times word wi appears in all documents of class ci.

$$\begin{aligned} \Pr(w_i|c) &= \frac{\text{number of times } w_i \text{ appears in docs of class c}}{\text{total number of words in class c in training dataset}} \\ &= \frac{count(w_i, D_c)}{\sum_{w' \in V} count(w', D_c)} \\ &= \frac{count(w_i, D_c)}{\sum_{d \in D_c} len(d)} \quad \text{more intuitive sum} \end{aligned}$$

[4]

To account for zero probabilities, that is, situation happens when fi is not in any training set of a given class ci, but is in a different class cj. If a likelihood term is 0, it results in the probability of the entire class to turn 0 regardless of the likelihoods of other features, which isn't accurate by any means. Thus we utilized add-one Laplace smoothing.

$$\begin{aligned} \Pr(w_i|c) &= \frac{count(w_i, D_c) + 1}{\sum_{w' \in V} (count(w', D_c) + 1)} \\ &= \frac{count(w_i, D_c) + 1}{\sum_{w' \in V} (count(w', D_c)) + |V|} \\ &= \frac{count(w_i, D_c) + 1}{\sum_{d \in D_c} (len(d)) + |V|} \quad \text{more intuitive sum} \end{aligned}$$

[4]

Sometimes Computers return to accuracy of specific decimal point and the product of probabilities occurring between 0 and 1 return very small decimals which will not be stored in memory and that contributes as 0. To account for floating-point underflow, we use ln's.

$$\hat{c} = \max_{c \in C} log(\Pr(c)) + log(\Pr(f_i, c))$$

[4]

III. LIMITATIONS OF NAÏVE BAYES CLASSIFIER

The first limitation is that the Naive Bayes Classifier considers a strong assumption regarding feature distribution, i.e. for a given output class, two features are independent of each other. Due to this, the result can be very bad. This independent assumption is called conditional independence. It is self-explanatory that the conditional independence assumption is hardly ever true in almost all real-world applications [10]. However, this is not as troublesome as people think, because the Naïve Bayes classifier can be optimal even if the assumption is broken [5] but its outcomes can be great even on account of sub-optimality. Another issue occurs due of information shortage. For any imaginable estimation, you have to measure probability esteem by a frequented methodology. This can result in probabilities going towards 0 or 1, which thus prompts numerical insecurities and more regrettable outcomes. For this problem, we have to smooth probabilities some way on your information, but you may contend that the subsequent classifier isn't naive any longer [6]. A third issue emerges for continuous features. It is common to utilize a binning method to make them discrete, yet in the event that you are not cautious you can discard a great deal of data. Plausibility is to utilize Gaussian appropriations for the probabilities [7].

IV. NEED FOR DATA PRE-PROCESSING

Sentiment analysis is never just about calculating sentiment of provided data; rather analyzing given data in an appropriate format happens to be of upmost importance. With efficiently processed data, we can achieve more accurate results. Thus, given a set of raw data sets, the first process in sentiment analysis is the pre-processing of that data. Pre-processing involves a set of techniques that increases value of the next phases of elaboration, in order to achieve better performances [8]. Two important operations involved in data pre-processing are:

A. Feature Extraction:

This phase consists of several techniques such as tokenization, stemming or lemmatization, stop word removal. Tokenization, also called as Part of Speech tagging assigns tags to each of the word in text and distinguishes a word into morphological category such as norm, verb, adjective, etc. Part of speech taggers are useful for explicit feature extraction in terms of accuracy [9]. Next step is to normalize the data obtained by stemming or lemmatization. The stemming process converts all the words into a root form called a stem. Stemming gives faster performance where accuracy is not the major concern. Lemmatization groups together different forms of word into a single one. Stemming removes inflected words only; on the other hand, Lemmatization replaces words with its base form. For example, the words like “cars” or “caring” are reduced to “car” in a stemming process whereas lemmatization reduces it to “car” and “care” respectively; hence lemmatization is considered to be more accurate [9]. Stop words are high frequency words such as “a”, “an”, “for”, “the”, “in”. The stop words removal reduces complexity of the data sets and thus key words can be recognized more easily by the automatic feature extraction techniques. Words to be eliminated are taken from a commonly available list of stop words. This can be implemented by using Machine learning toolkits such as NLTK, GATE and WEKA [9].

B. Feature Selection:

Few important reasons to use feature selection are as follows: It enables the algorithms to train faster, complexity of model is reduced. If the right subset is decided then it accelerates the accuracy of the model and also reduces over fitting [8]. Use of N-gram techniques compared to a single word since it has an advantage that there occur some dependencies between certain words and importance placed on individual phrases or words [11]. Different techniques involved in feature selection are filter methods, wrapper methods and embedded methods [8]. In filter methods, features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable. The feature with the most importance is made a part of the feature space. The downside is that multi co-linearity is not addressed in this method. Various statistical tests involved in filter selection are Pearson’s Correlation, LDA, ANOVA and Chi-Squared Test [8]. In wrapper method, subset of features is used and models are trained using them. Wrapper methods involve Forward Selection, Backward Selection and Exhaustive Selection. Computationally, these methods are very expensive but do yield satisfactory results [8]. Qualities of filter and wrapper methods are combined in embedded methods. It’s implemented by algorithms that have their own built-in feature selection methods. Some of the most popular examples of these methods are LASSO and RIDGE regression which have inbuilt penalization functions to reduce over fitting [8].

V. PROPOSED MODEL

If your classification model has hundreds or thousands of features, as is the case with text categorization, it is a good chance that many features are low information. These are the features common throughout all classes and hence contribute little information to the process of classification. They are harmless at an individual level, but can decrease performance of the classifier in aggregate, low information features. By removing noisy data, the elimination of low information features gives the model a

clarity. It can save the model from the curse of dimensionality and overfitting. If only the higher information features are used, it can increase performance while also decreasing the model size, resulting in less memory usage along with faster training.
In order to improve efficiency, redundancy has to be dealt first. For that purpose, one needs to calculate "information gain" for each word appearing in text documents. Information gain basically measures how often a particular word/feature has appeared in a class compared to how common it is to other existing classes. A feature that occurs primarily in positive dataset and rarely in negative dataset is high information. For e.g. "awesome" in a movie review is a strong indicator of review being positive which makes the word high information. It is noticeable that most informative feature is unaffected which makes sense because purpose is to make use of informative feature and ignore the rest. Information gain is calculated using Chi Square metric. NLTK includes this property in the BigramAssocMeasures class in the metrics package. To use it, we need to calculate a few frequencies for each word: its overall frequency and its frequency within each class. This is done with a FreqDist for overall frequency of words, and a ConditionalFreqDist where the conditions are the class labels. Once we have those numbers, we can score words with the BigramAssocMeasures.chi_sq function, then sort the words by score and take the top 10000. We then put these words into a set, and use a set membership test in our feature selection function to select only those words that appear in the set. Now each file is classified based on the presence of these high information words. Classification on the basis of these high information words will result in more accurate prediction.

VI. CONCLUSION

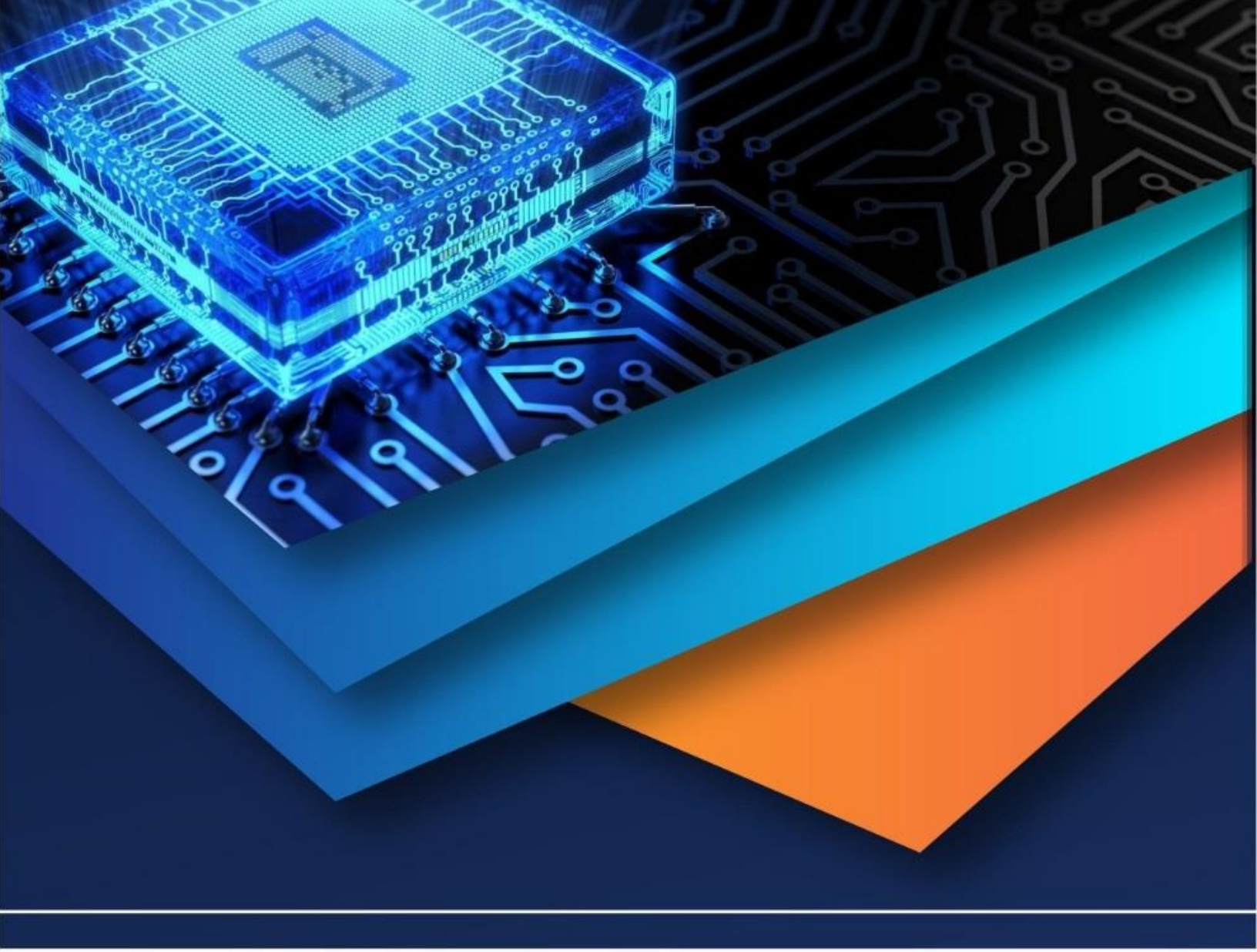
The first and foremost lesson in improving the Naïve Bayes classifier is to present the more efficient feature set and also to provide more and more context to the classifier. The efficient features space will help the classifier to make a better probability calculation and also help to lean towards one of the classes to give a better polarity. Also, more context will help the classifier to classify difficult phrases and return an accurate sentiment decision. Feature selection can help in achieving the targeted results while reducing the complexity.

ACKNOWLEDGMENT

We would like to acknowledge our mentor Mrs. Harshali Patil who has been a constant source of knowledge and support right from the start. This paper would not have been possible without her guidance.

REFERENCES

[1] https://triton.ml/blog/sentiment-analysis
[2] Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. Proceedings of ECML '98
[3] Yang, Y., & Liu, X. (1999). A re-examination of text categorization methods. Proceedings of SIGIR '99.
[4] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/
[5] Domingos, P. and Pazzani, M., 1997. On the optimality of the simple Bayesian classifier under zero-one loss. Machine learning, 29(2-3), pp.103-130.
[6] https://www.quora.com/What-are-the-disadvantages-of-using-a-naive-bayes-for-classification
[7] John, G.H. and Langley, P., 1995. Estimating continuous distributions in Bayesian classifiers. In Proceedings of the Eleventh conference on Uncertainty in artificial intelligence (pp. 338-345). Morgan Kaufmann Publishers Inc.
[8] http://ceur-ws.org/Vol-1748/paper-06.pdf
[9] http://www.statmt.org/OSMOSES/FeatureEx.pdf
[10] The Optimality of Naive Bayes. Harry Zhang. Faculty of Computer Science. University of New Brunswick. Fredericton, New Brunswick, Canada.E3B5A3
[11] Yousefpour, A., Ibrahim, R., Nuzly, H., & Hamed, A. (2014). A Novel Feature Reduction Method in Sentiment Analysis. International Journal of Innovative Computing, 4(1), 34–40.
[12] https://streamhacker.com/2010/06/16/text-classification-sentiment-analysis-eliminate-low-information-features/



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY