International Journal of Computing Academic Research (IJCAR)
ISSN 2305-9184, Volume 8, Number 5 (October 2019), pp.75-92
© MEACSE Publications
http://www.meacse.org/ijcar

Performance of Naïve and Complement Naïve Bayes Algorithms Based on Accuracy, Precision and Recall Performance Evaluation Criterions

Berna Seref and Erkan Bostanci
Computer Engineering Department, Ankara University, Ankara, Turkey

Abstract

Machine learning algorithms are one of the techniques that used to classify texts. There are two types of learnings which are supervised and unsupervised. In supervised learning, data samples have label information and samples must be assigned to the label which they are similar. In unsupervised learning, labels are not known. Similar samples are grouped and then labels are assigned to these groups. In this paper, Naïve Bayes and Complement Naïve Bayes Algorithms are analyzed. Naïve Bayes and Complement Naïve Bayes Classification Algorithms are used to classify texts because of their fast and easy implementation. On the other hand, both of them has some advantages and disadvantages. Complement Naïve Bayes Algorithm is formed to complete some disadvantages of Naïve Bayes Algorithm such as skewed bias data. In this paper, performance of Naïve Bayes and Complement Naïve Bayes Algorithms are compared changing on smoothing, normalization and weight parameters with different size of the training and test datasets. Dataset is scaled up by 2, 4 and 8. Experiments are carried on changing normalization parameter, weight parameter and smoothing parameter with different size of the dataset. All experiments for all parameters are done for 10 times, then average of the results is calculated and saved. Performance of both algorithms are observed on accuracy, precision and recall performance evaluation formulas with different data sizes. Their implementation is carried on Hadoop platform and with Mahout library which are for big data processing. As a result, it is observed that performance of Complement Naïve Bayes is better than performance of Naïve Bayes.

Keywords:Naïve Bayes, Complement Naïve Bayes, accuracy, precision, recall, big data.

Introduction

Machine learning is a technology that learn from existing data and used for knowledge discovery and predictive analytics. To make a good decision, it is adopted for spam filtering [1,2], advertisement targeting [1,3], computer vision [1,4], and bioinformatics [1,5], etc. applications. There are two types of machine learning algorithms: supervised and unsupervised. In supervised learning, data samples have label information and samples must be assigned to the label which they are similar. Classification is based on supervised learning. In unsupervised learning, labels are not known. Similar samples are grouped and then labels are assigned to these groups. Clustering is based on unsupervised learning [1].

Big data is defined with volume, velocity and variety. While volume refers to the size of the data, velocity refers to the data must be processed as soon as possible to prevent synchronization problems. After data is proceed, it must have a variety and help to make good decisions.

Apache Hadoop is able to handle big data challenges which are volume, velocity and variety. It consists of two main elements: MapReduce and Hadoop Distributed File System (HDFS). MapReduce is responsible for processing data in parallel. It consists of two important steps: map and reduce. In map step, data is converted to key/value pair. In reduce step, data is collected, some computations are performed and single value is created. HDFS leads high-bandwidth computation and distributed low-cost storage. It consists of two major elements: NameNode and DataNodes. NameNode is responsible for managing all of the file system metadata and DataNode is responsible for storing the blocks of datasets [1].

Machine learning algorithms are one of the techniques that used to classify texts and different type of the data types. Naive Bayes and Complement Naive Bayes Algorithms are one of these algorithms. Naive Bayes Algorithm is the most popular one because of its easy and fast implementation.

Naive Bayes Algorithm is defined as "the punching bag of classifiers" [6,7]. Other classification algorithms can give less erroneous results but they are too complex and slow.

Naive Bayes Algorithm has some problems that cause poor performance such as poor prediction ability, selecting poor weights for the decision boundary, assuming features are independent and not modelling texts well. To overcome these systematic problems and improve classification accuracy, some techniques are introduced [7].

In this study, performances of Naïve Bayes and Complement Naïve Bayes Algorithms are compared based on changing smoothing, normalization, weight parameters and data size. Average accuracy, average weighted precision and recall are chosen as a performance evaluation criterion.

There are some studies which are similar to our study. While some of them compare performance of Naïve Bayes, Complement Naïve Bayes and other algorithms, some of them compare performance of Naïve Bayes with other algorithms. We get similar results with the studies in the literature.

There are some studies which are similar to our study. A. M. Kibriya et al. [8], L. Jiang et al. [9], A. Anagaw and Y.L. Chang [10], K. Komiya et al. [11] and V. Keiser and T. G. Dietterich [12] compare performance of Naïve Bayes, Complement Naïve Bayes and other algorithms. For example, while A. M. Kibriya et al. [8] compare performance of Multinomial Naive Bayes, Weight-Normalized Complement Naive Bayes classifier (TWCNB) and support vector machines and improve Naïve Bayes Algorithm performance using TFIDF scores, L. Jiang et al. [9] compare accuracy performance of multinomial naive Bayes (MNB), complement naive Bayes (CNB), the one-versus-all-but-one (OVA) model, Local Weighted Multinominal Bayes (LWMNB), Local Weighted Complement Naïve Bayes (LWCNB) and Local Weighted one-versus-all-but-one (LWOVA). On the other hand, A. Anagaw and Y.L. Chang [10] compare classification performance of Complement Naïve Bayes with Naïve Bayes Algorithm, K. Komiya and others [11] compare performance of Negation Naive Bayes (NNB) with the performance of Naïve Bayes and Complement Naïve Bayes. In addition to them, V. Keiser and T. G. Dietterich [12] compare classification performance of Bernoulli Naïve Bayes, Multinomial Naïve Bayes, Transformed Weight-Normalized Complement Naïve Bayes, Term Frequency – Inverse Document Frequency counts, Online Passive Aggressive algorithms, and Linear Confidence Weighted classifiers. In general, they get similar results. For example; A. M. Kibriya et al. [8] claims that performance of Naïve Bayes and Complement Naïve Bayes depends on dataset and their performance is close to each other. They propose using support vector machine if the performance evaluation method is accuracy. L. Jiang et al. [9] rank test on classification accuracy is as follows: LWOVA, LWCNB, LWMNB, OVA, CNB and MNB. A. Anagaw and Y.L. Chang [10] observe that Complement Naïve Bayes Algorithm shows better performance based on both computational time and accuracy performance. While K. Komiya et al. [11] propose Negation Naive Bayes (NNB) if data is non-uniformly distributed, V. Keiser and T. G. Dietterich [12] propose Linear Confidence Weighted classifiers.

There are some studies which compare performance of Naïve Bayes Algorithm with other algorithms. For example; A. Goyal and R. Mehta [13] compare classification performance of Naïve Bayes Algorithm and J48 Algorithm, and A. Ashari and et al. [14] compare classification performance of Naïve Bayes, Decision Tree and k-Nearest Neighbor Algorithms. While N. Friedman and M. Goldszmidt [15] single out a method which is called Tree Augmented Naive Bayes (TAN) and compare classification accuracy of NBC (Naïve Bayesian Classifier), Unsup (Unsupervised Bayesian networks learned using the MDL (minimal description length) score), TAN (Tree Augmented Naive Bayes), TANS (Smoothed TAN networks), C4.5 (the decision-tree classifier), SNBC (the Selective Naive Bayesian Classifier), R. Nithya et al. [16] compare classification performance of Bayes Net, Naive Bayes, Naive Bayes Multinomial Text. In addition to them, H.Y. Mussa et al. [17] propose Tapered Naïve Bayes (TNB) classification algorithm and compare performance of it by Laplacian Corrected Modified Naïve Bayes (LCMNB) and Standard Naïve Bayes (SNB) Algorithms. After the experiments, while some studies [14,16] observe that performance of Naïve Bayes Algorithm is better, the other studies propose J48 Algorithm [13] and TNB [17].

Difference of our study from the studies in the literature is that we analyze effect of changing smoothing, normalization and weight parameters and data size on performance of Naïve Bayes and Complement Naïve Bayes Algorithms according to average accuracy, average weighted recall and precision performance evaluation criterions. After that, we compare performance of these algorithms based on the mentioned performance evaluation criterions.

In this paper, firstly, information about Naive Bayes and Complement Naive Bayes Algorithms are given. Secondly, reasons of Naive Bayes's poor performance are explained and some solutions are proposed. After that, material and methods are described, then, results are shown. Finally, paper is concluded.

Naïve Bayes and Complement Naïve Bayes Algorithms

Naïve Bayes Algorithm

Naive Bayes is one of the machine learning algorithms used for classification. It is combination of Bayes' Theorem and the attribute independence assumption [18].
For classification, assume that $c \epsilon \{1,2,\ldots,m\}$where c is fixed number of classes. The parameter vector for a class c is$\vec{\theta}_c$and defined as $\vec{\theta}_c = \{\theta_{c1}, \theta_{c2,\ldots}\ \theta_{cn}\}$where n isthe size of the vocabulary. θ_{ci}means the probability that word $i$occurs in that class and $\Sigma_i \theta_{c_i} = 1$.Likelihood of a document is given in Eq. (1) [7].

$$p(d|\vec{\theta}) = \frac{(\Sigma_i f_i)!}{\Pi_i f_i!} \Pi_i (\theta_{ci})^{f_i} \qquad (1)$$

In Eq. (1), f_i means frequency count of word i in document d. Minimum error classification rule [7,19] is given in Eq. (2).

$$I(d) = argmax_c\left[log\, p(\vec{\theta}_c) + \textstyle\sum_i f_i\ log\ \theta_{ci}\right](2)$$
$$= argmaxc[bc + \textstyle\sum_i fiwci](3)$$

In Eq. (2) and Eq. (3), while b_c means the threshold term, wc_i means the class c weight for word i. $p(\vec{\theta}_{c)}$ is prior distribution over the set of classes [7].
$\theta\hat{}_{ci}$ is defined in Eq. (4) [7,20]:

$$\theta\hat{}_{ci} = \frac{Nci+ai}{Nc+a} \qquad (4)$$

In Eq. (4),whileN_{ci} means the number of times word i appears in the documents in class$c$,$N_c$means total number of word occurrences in class c. For $word\,i\alpha_i$is imagined occurrences to smooth the estimate in order to prevent zero probability. $\alpha$is sum of α_i[7].
Multinomial Bayes Classifier rule is defined in Eq. (5) [7].

$$IMNB(d) = argmax_c\left[\log \hat{p}(\theta_c) + \sum_i f_i \log \frac{Nci+ai}{Nc+a}\right](5)$$

In Eq. (5),$\hat{p}(\theta_c)$ is the class prior estimate.

Complement Naive Bayes Algorithm

In Multinomial Naive Bayes, to estimate weights, only a single class c is used. On the other hand, in Complement Naive Bayes Algorithm, all training data of the classes except c class is used [7]. Eq. (6) shows the formula of Complement Naive Bayes Algorithm rule [7].

$$ICNB(d) = argmax_c\left[\log p(\vec{\theta}_c) - \sum_i f_i \log \frac{N\tilde{c}i+ai}{N\tilde{c}+a}\right](6)$$

In Eq. (6), while$N\tilde{c}i$means number of times word i occurred in documents in classes other than c, $N\tilde{c}$means total number of word occurrences in classes other than c[7].

Performance of Naïve Bayes Algorithm

Naive Bayes Classification Algorithm has some problems about its performance. Two of the problems are skewed bias data and assuming features are independent.
Skewed bias data means that when number of training data of a class is more than the others, classifier choses this class. The reason of it that weights are shrinked for the other classes and poor weights are chosen for the decision boundary. However, the problem is solved with Complement Naive Bayes Algorithm. This algorithm uses all training data of the classes except class $c$training samples [7].
Another problem of Naive Bayes Algorithms is assuming features are independent. As a result, weight magnitude for classes with strong word dependencies are larger than classes with weak word dependencies. With the aim of keeping classes with more dependencies from dominating, weights are normalized [7].
In this study, it is aimed to overcome these problems by using Complement Naïve Bayes Algorithm and normalizing weights by *lnorm* or *n1* normalization parameters.

Experimental Design

Dataset

In this study, performance of Naive Bayes and Complement Naive Bayes Algorithms are compared on changing smoothing, normalization and weight parameters with different size of the training and test datasets using Mahout library on Hadoop platform.
As a training set "20 Newsgroups" dataset [21] is used. This dataset has 20 different newsgroups and in total, it has 18846 documents. It is scaled up by 2, 4 and 8. As a result, datasets with the size of 37692, 75384 and 150768 are created.

Experiments

Data is converted to sequence file format, then, it is converted to vector file format using a normalization parameter and weight parameter. 70% of vector file is defined as training set and 30% of the vector file is defined as test set randomly. As a smoothing parameter, a is defined 0.1. Training set is trained using Naïve Bayes Algorithm. After that, it is tested. Accuracy, recall, precision values are saved. Then, new training and test sets are defined randomly and new training set is trained and system is tested. Results are saved. These experiments are done for ten times. After that, average of results are calculated. After saving average results, smoothing parameter is increased by 0.1.Then, new training and test sets are defined randomly again and the experiments are carried out for ten times for new smoothing parameter. The same experiments are carried on again until the smoothing parameter is 1. After carrying on the experiments for 0.9 smoothing parameter value, sequence file is converted to vector file using new normalization and weight parameter. After that,a is defined 0.1 and the same experiments are carried on again.
The same experiments are carried on for lnorm-tf, n1-tf, n1-tfidf normalization and weight parameters and datasets with 18846, 37692, 75384,150768 documents for Naïve and Complement Naïve Bayes Algorithms.

Results

Results are given below:

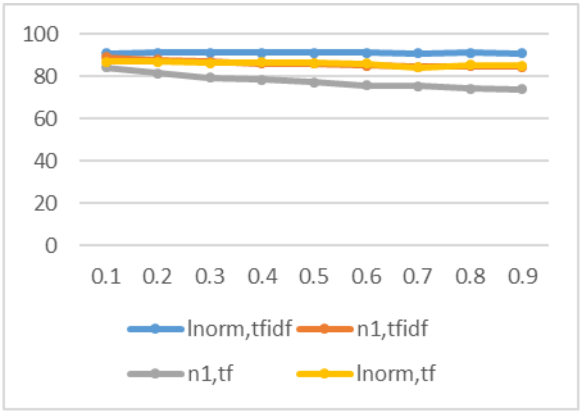

Fig. 1. Average accuracy results for normalization, smoothing and weight parameters for 18846 documents using Naive Bayes Algorithm.

In Fig. 1, while x axis represents smoothing parameter, y axis represents average accuracy level. As it can be seen in Fig. 1, the best performance belongs to *lnorm-tfidf* results. *Lnorm-tfidf* results increases until 0.5 smoothing parameters. It starts to decrease for 0.6 and 0.7 smoothing parameters. For 0.8 smoothing parameter, accuracy value increases but than it decreases again. Results for *n1-tfidf* and *lnorm-tf* are very close to each other. *N1-tfidf* results decreases when smoothing parameter isincreased until 0.7. After 0.7 it increases, then it decreases again. *N1-tf* results decreases when smoothing parameter increases. *Lnorm-tf* results increases then decreases when smoothing parameter is increased.

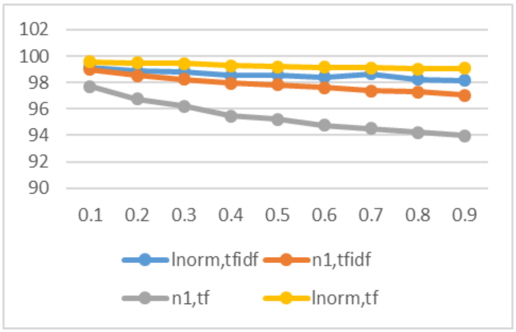

Fig. 2. Average accuracy results for different normalization, smoothing and weight parameters for 150768 documents using Naive Bayes Algorithm.

In Fig. 2, while x axis represents smoothing parameter, y axis represents average accuracy level. As it can be seen in Fig. 2, when Fig. 1 and Fig. 2 are compared, it is observed that the best performance belongs to dataset with 150768 documents for all types of the normalization and weightparameters. In Fig. 2, when smoothing parameter is increased, accuracy decreases after that it increases again for *lnorm-tfidf*. Accuracy tendency is not stable. For *n1-tf*, when smoothing parameter is increased, accuracy level decreases. For *n1-tfidf*, when smoothing parameter is increased, accuracy level decreases. For *lnorm-tf*, when smoothing parameter is increased, accuracy level decreases except 0.9 smoothing parameter.

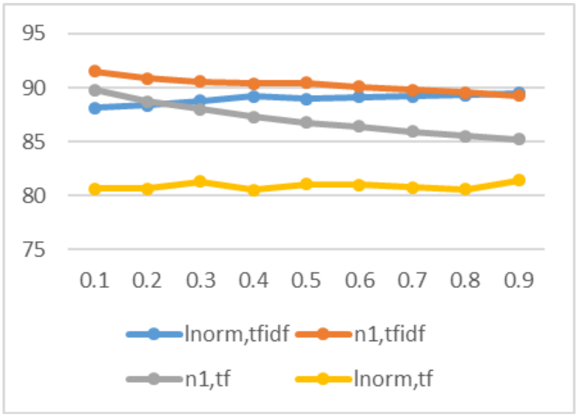

Fig. 3. Average accuracy results for different normalization, smoothing and weight parameters for 18846 documents using Complement Naive Bayes Algorithm.

In Fig. 3, while x axis represents smoothing parameter, y axis represents average accuracy level. In Fig. 3, it is observed that when smoothing parameter is increased, accuracy level increases for *lnorm-tfidf* except 0.5 smoothing parameter. When smoothing parameter is increased, accuracy level decreases for *n1-tfidf* except 0.5 smoothing parameter. When smoothing parameter is increased, accuracy level decreases for *n1-tf*. When smoothing parameter is increased, accuracy level decreases, after that, it starts to increase for *lnorm-tf*. Accuracy level tendency is not stable.

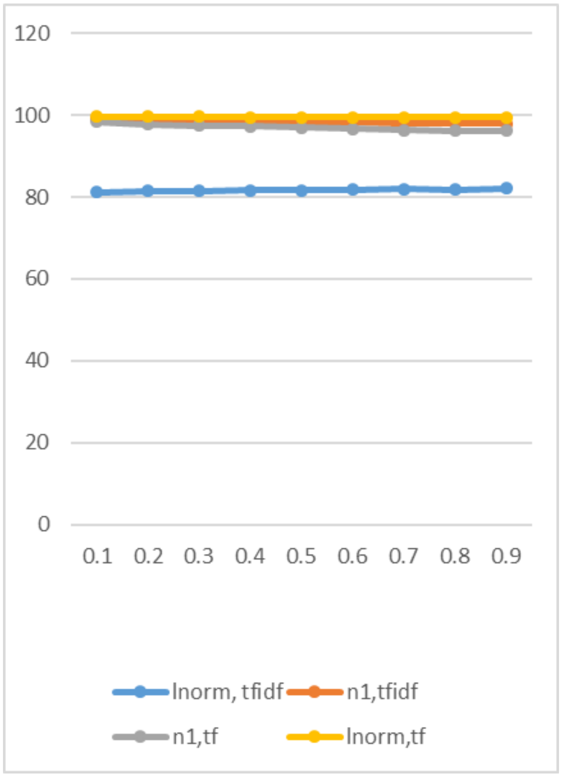

Fig. 4. Average accuracy results for different normalization, smoothing and weight parameters for 150768 documents using Complement Naive Bayes Algorithm.

In Fig. 4, while x axis represents smoothing parameter, y axis represents average accuracy level. In Fig. 4, it is observed that values for *n1-tf*, *n1-tfidf* and *lnorm-tf* are very close to each other. When smoothing parameter is increased, accuracy values for the dataset with 150768 increases. For *lnorm-tfidf*, until 0.5 smoothing parameter, accuracy values increases. For0.5 smoothing parameter it decreases. For 0.6 and 0.7 parameters accuracy increases but for 0.8 parameter it decreases again. For *n1-tfidf*, it decreases until 0.8 smoothing value. For 0.8 smoothing value, accuracy value increases, then it decreases for 0.9 smoothing parameter value. For *n1-tf*, accuracy values decrease. For *lnorm-tf*, accuracy values decrease until 0.5 smoothing parameter. When it is 0.5, accuracy value increases. After that, we can not observe stable tendency between smoothing parameters and accuracy values.

Performance of Naive Bayes and Complement Naive Bayes Algorithms differs each other on behalf of normalization parameter, weight and smoothing parameters. The best performance of them are given in below Figures.

Figure 5 shows the best accuracy performance based on different size of the datasets for Naive Bayes Algorithm.

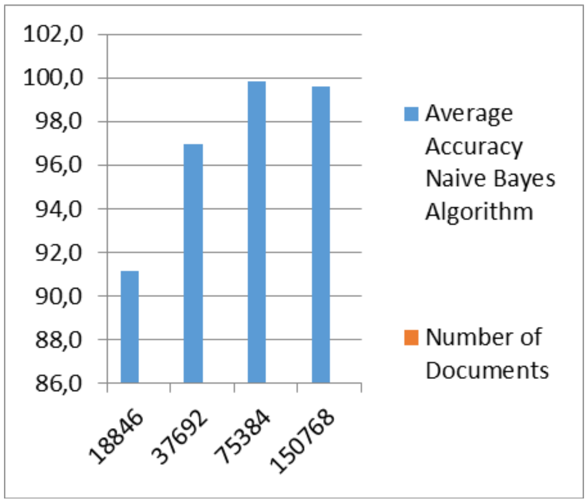

Fig. 5. The best average accuracy performance of Naive Bayes Algorithm based on different size of the datasets.

As it can be seen in Fig. 5, increasing size of the dataset increases the accuracy. However, the best performance belongs to the size of 75384 instead of 150768 documents because of overfitting.

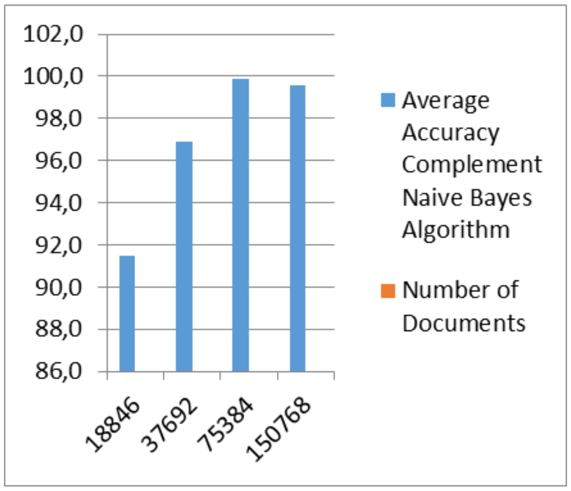

Fig. 6. The best average accuracy performance of Complement Naive Bayes Algorithm based on different size of the datasets.

When performance of Naive Bayes and Complement Naive Bayes Algorithms are compared using Fig.5 and Fig. 6, it is observed that performance of Complement Naive Bayes is better.

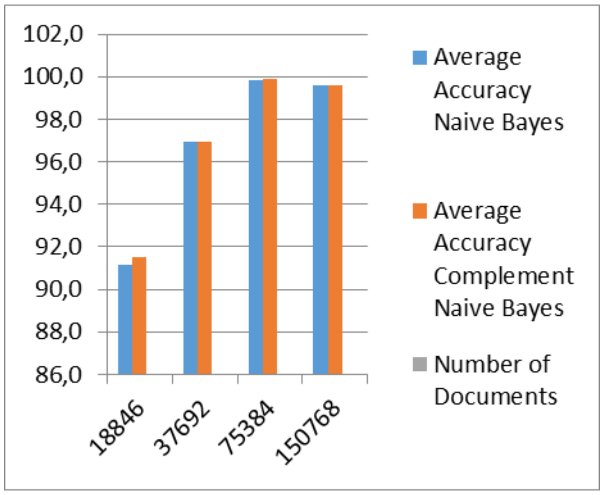

Fig. 7. The best average accuracy performance of Naive Bayes and Complement Naive Bayes Algorithms with different size of the datasets.

As it can be seen in Fig.7, except the dataset with 37692 documents, average accuracy performance of Complement Naive Bayes Algorithm is better than Naive Bayes Algorithm for the reason that Complement Naive Bayes algorithm uses all thetraining set except the training data in the class and prevent skewed data bias.

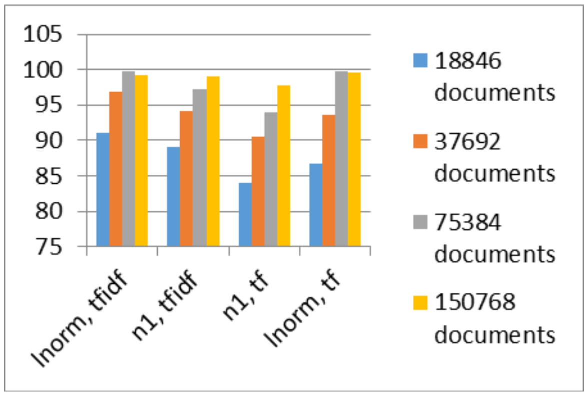

Fig. 8. The best average accuracy performance of datasets based on normalization and weight parameters for Naive Bayes Algorithm.

In Fig. 8, it is observed that the smallest dataset gives the best performance when normalization parameter is *lnorm* and weight parameter is *tfidf*. Dataset with 37692, 75384 documents have the same tendency with the dataset of 18846 datasets. The biggest dataset with 150768 documents shows the best performance when normalization parameter is *lnorm* and weight parameter is *tf*.

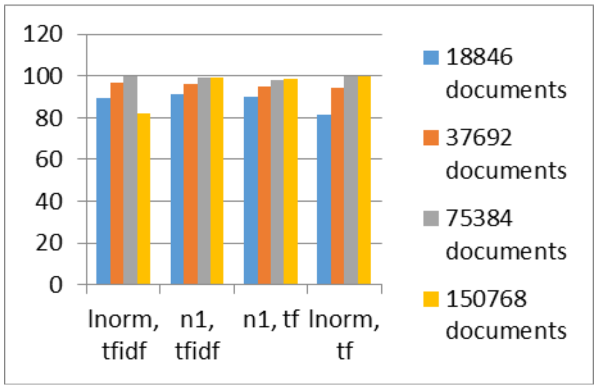

Fig. 9. The best average accuracy performance of datasets based on normalization and weight parameters for Complement Naive Bayes Algorithm.

In Fig. 9, it is observed that 18846 documents show the best performance when normalization parameter is *n1* and weight parameter is *tfidf*. Datasetsize with 37692 documents shows the best performance when normalization parameter is *lnorm* and weight parameter is *tfidf*. Dataset sizes with 75384 and 150768 show the best performance when normalization parameter is *lnorm* and weight parameter is *tf*.
When the best performances of the algorithms are compared, it is observed that Complement Naive Bayes Algorithm is better than Naive Bayes Algorithm.

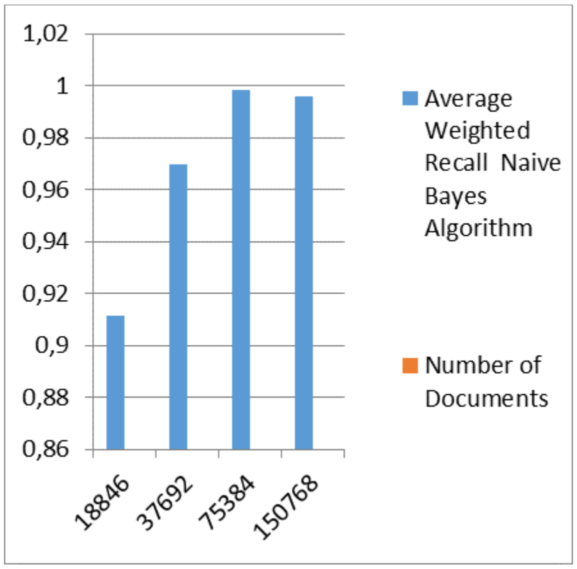

Fig. 10. The best average weighted recall performance of Naive Bayes Algorithm based on different size of the datasets.

In Fig. 10, it is observed that increasing size of the dataset improves the performance. The best average weighted recall value belongs to dataset with the size of 75384 documents. After this dataset, recall value starts to decrease. The reason of it that, model starts to predict worse because of overfitting.

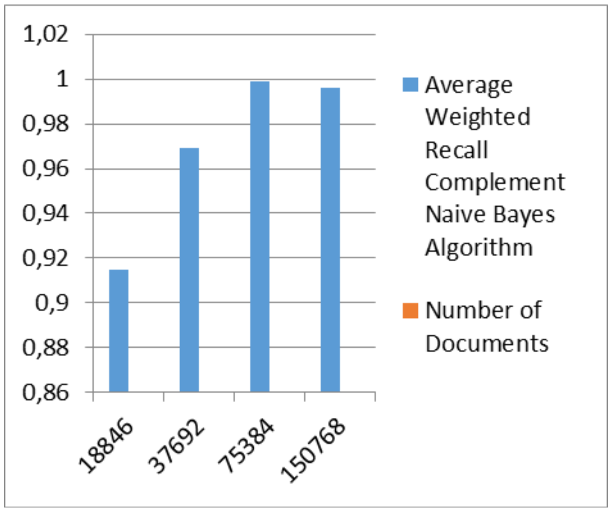

Fig. 11. The best average weighted recall performance of Complement Naive Bayes Algorithm based on different size of the datasets.

In Fig. 11, it is observed that the best weighted recall value belongs to dataset with 75384 documents and increasing size of the dataset improves the performance in general.

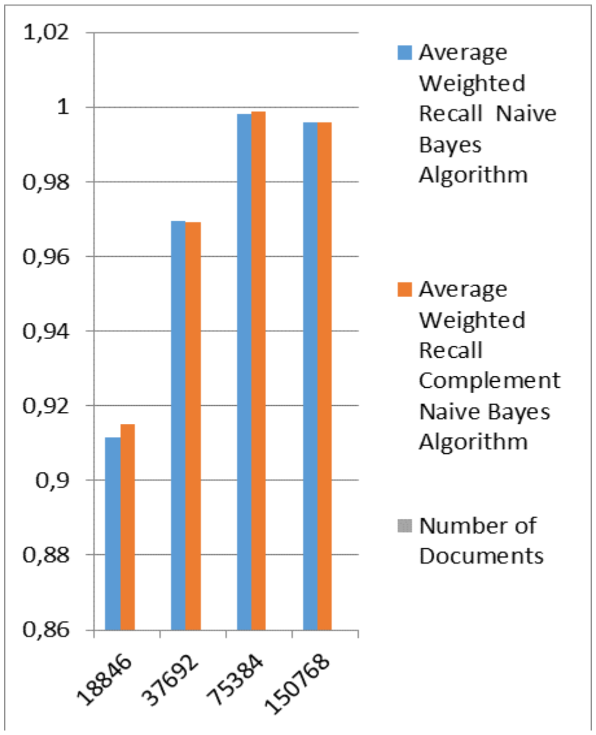

Fig. 12. The best average weighted recall performance of Naive Bayes and Complement Naive Bayes Algorithms with different size of the datasets.

In Fig. 12, it is observed that average weighted recall performance of Complement Naive Bayes Algorithm is better than Naive Bayes Algorithm's performance except the result for dataset with 37692 documents. The best performance is with the dataset which has 75384 documents and it is too close to the optimal value which is 1.

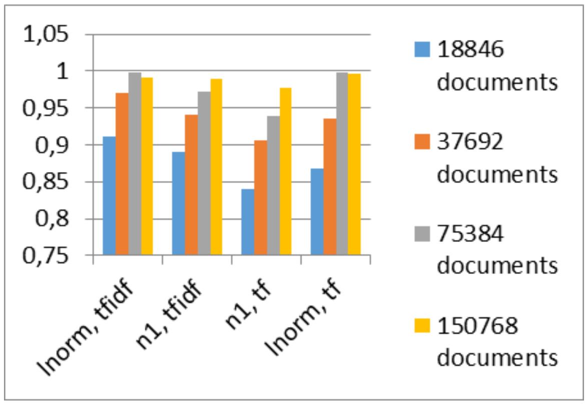

Fig. 13. The best average weighted recall performance of datasets based on normalization and weight parameters for Naive Bayes Algorithm.

In Fig. 13, it can be seen that the smallest dataset gives the best performance when normalization parameter is *lnorm* and weight parameter is *tfidf*. Datasets with 37692 and 75384 shows the best performance when normalization parameter is *lnorm* and weight parameter is *tfidf*. The biggest size of the dataset with 150768 documents shows the best performance when normalization parameter is *lnorm* and weight parameter is *tf*.

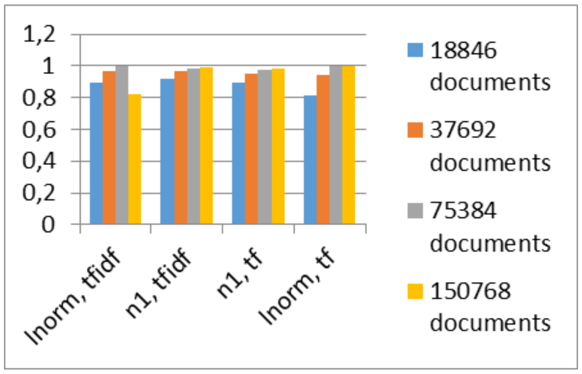

Fig. 14. The best average weighted recall performance of datasets based on normalization and weight parameters for Complement Naive Bayes Algorithm.

In Figure 14, it is observed that the smallest dataset gives the best performance when normalization parameter is *n1* and weight parameter is *tfidf*. Dataset with 37692 documents gives the best performance when normalization parameter is *lnorm* and weight parameter is *tfidf* as Naive Bayes Algorithm's result. Datasets with 75384 and 150768 documents shows the best performance when normalization parameter is *lnorm* and weight parameter is *tf*. For 150768 documents Naive Bayes and Complement Naive Bayes Algorithms give the best performance when the same normalization and weight parameters are used.

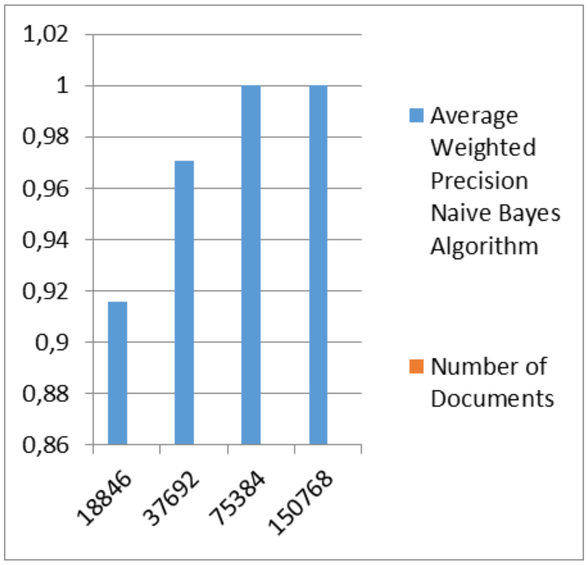

Fig. 15. The best average weighted precision performance of Naive Bayes Algorithm based on different size of the datasets.

In Fig. 15, it is observed that increasing size of the dataset improves the performance. Datasets with 75384 and 150768 documents give the best performance which is 1.

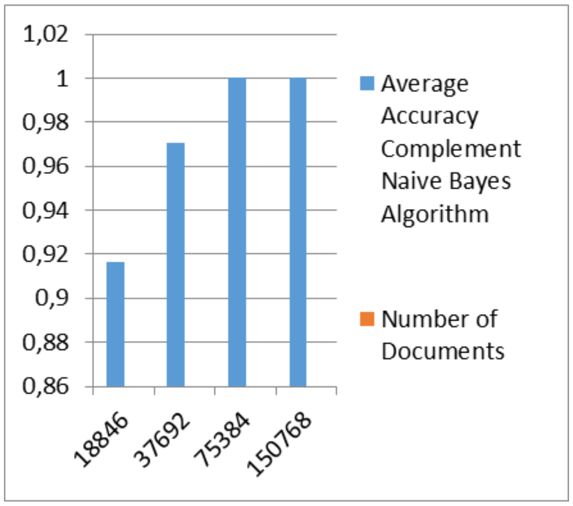

Fig. 16. The best average weighted precision performance of Complement Naive Bayes Algorithm based on different size of the datasets.

In Figure 16, it is observed that increasing size of the dataset improves the performance. Datasets with 75384 and 150768 documents give the best performance which is 1 as Naive Bayes Algorithm's result.

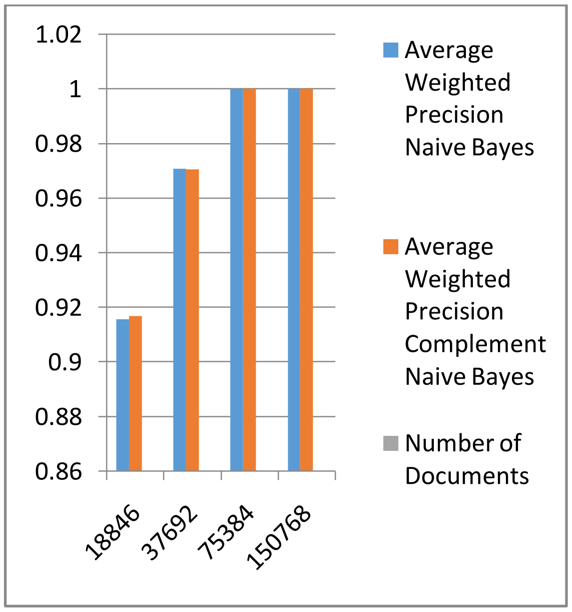

Fig. 17. The best average weighted precision performance of Naive Bayes and Complement Naive Bayes Algorithms with different size of the datasets.

In Fig. 17, it is observed that only for dataset with 37692 documents gives the best performance when Naive Bayes Algorithm is used. Averageweighted precision for the datasets are too close when Naive Bayes and Complement Naive Bayes Algorithms are used. For 75384 and 150768 documents the weighted precision values are the same which is equal to 1.

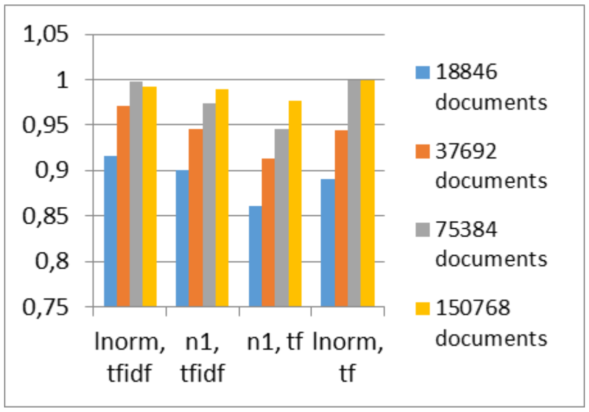

Fig. 18. The best average weighted precision performance of datasets based on normalization and weight parameters for Naive Bayes Algorithm.

In Fig. 18, it is observed that dataset with 18846 and 37692 documents gives the best performance when normalization parameter is *lnorm* and weight parameter is *tfidf*. Datasets with 75384 and 150768 documents give the best performance when normalization parameter is *lnorm* and weight parameter is *tf*. As a result, the best performances are obtained when normalization parameter is *lnorm*.

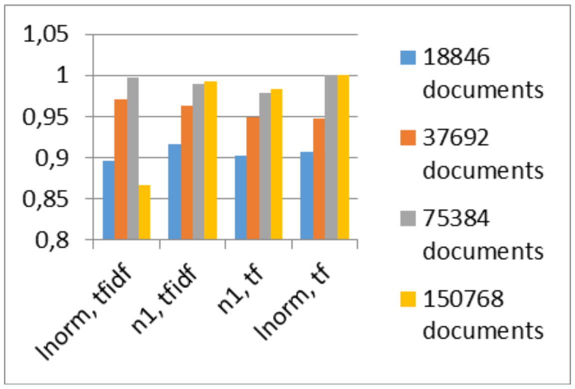

Fig. 19. The best average weighted precision performance of datasets based on normalization and weight parameters for Complement Naive Bayes Algorithm.

In Fig. 19, it is observed that dataset with 18846 documents give the best performance when normalization parameter is *n1* and weight parameter is *tfidf*. For dataset with 37692 documents shows the best performance when normalization parameter is *lnorm* and weight parameter is *tfidf* as Naive Bayes Algorithm. Datasets with 75384 and 150768 documents give the best performance when normalization parameter is *lnorm* and weight parameter is *tf* as Naive Bayes results.

Performance of Naive Bayes and Complement Naive Bayes Algorithms are very close to each other according to the results given in Table 1 and Table 2. While Table 1 has t test results, Table 2 has f test results. In Table 1 and Table 2, p, critical one tail and statistical values for average accuracy, average weighted recall and average weighted precision criterions are given.

Table 1.P, Critical One Tail And Statistical Values For T Test

	P value	**Critical one tail value**	**Statistical value**
Average accuracy	0.45575314	1.75305036	0.113029162
Average weighted recall	0.44361690	1.75305036	0.144236431
Average weighted precision	0.391029643	1.75305036	-0.28165141

Table 2.P, Critical One Tail And Statistical Values For F Test

	P value	**Critical one tail value**	**Statistical value**
Average accuracy	0.24517807	0.69549154	0.07499835

Average weighted recall	0.25128596	0.70262898	0.09577654
Average weighted precision	0.47219	1.03731178	-0.1710769

When t test results are analyzed, for average accuracy, average weighted recall and average weighted precision criterions, statistical value is smaller than critical one tail value. P value is bigger than 0.05. As a result, it can be said that these two algorithms are similar and there is no statistically significant different between groups. When f test results are analyzed, for average accuracy, average weighted recall and average weighted precision criterions, statistical value is smaller than critical one tail value. P value is bigger than 0.05. As a result, it can be said that these two algorithms are similar and there is no statistically significant different between groups. According to results in Table 1 and Table 2, performance of Naïve Bayes and Complement Naïve Bayes Algorithms are similar and there is no statistically significant different between performances of these algorithms.

Conclusions

In this study, it is aimed to compare performance of Naïve Bayes and Complement Naïve Bayes Algorithms for different smoothing, normalization, weight parameters and data size. To achieve that, "20 Newsgroups" dataset is scaled up by 2, 4 and 8. Experiments are carried on Hadoop platform using Mahout machine learning library. As a result, average accuracy, average weighted recall and average weighted precision performance of Naive Bayes and Complement Naive Bayes Algorithms are compared. It is observed that;

- Average accuracy values for Complement Naive Bayes Algorithms is better than performance of Naive Bayes Algorithm except the dataset with 37692 documents.
- Performance of average weighted recall for Complement Naive Bayes is better than Naive Bayes Algorithm except the dataset with 37692 documents.
- Performance of average weighted precision for Naive Bayes and Complement Naive Bayes Algorithms are very close to each other. For the datasets with 75384 and 150768 documents, theperformance of the algorithms is the same which is 1. While for the dataset with 18846 documents performance of Complement Naive Bayes is better, for the dataset with 37692 documents, performance of Naive Bayes Algorithm is better.
- In general, the best performances are obtained when smoothing parameter is 0.1
- In general, *lnorm* normalization parameter gives the best performance for Naive Bayes and Complement Naive Bayes Algorithm for all size of the datasets.
- In general, *tfidf* weight parameter improves performance of Naïve Bayes and Complement Naïve Bayes Algorithms.

- According to t test and f test results, performances of Naïve Bayes and Complement Naïve Bayes Algorithms are similar and there is no statistically significant difference between performances of these algorithms.

This paper can improve performance of the other researches for the reason that it compares performances of Naïve Bayes and Complement Naïve Bayes based on different smoothing, normalization and weight parameter and different size of the datasets. It shows tendency of the average accuracy according to different smoothing parameter for both of the algorithms. It observes normalization and weight parameter which gives the best average accuracy, average weighted recall and average weighted precision results for both of the algorithms. In addition to them, it studies on effect of the different size of the datasets for the performances.

References:

[1] S. Zheng, "Naive Bayes Classifier: A Mapreduce Approach", Master of Science, North Dakota State University, 2014

[2] T. Guzella and W. Caminhas., "A review of machine learning approaches to Spam filtering", Expert Systems With Applications 36(7), pp. 10206–10222, 2009

[3] W. Yih, J. Goodman and V. Carvalho, "Finding advertising keywords on web pages", Proceedings of the 15th international conference on World Wide Web, 2006

[4] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection", Proceedings of the 9th European conference on Computer Vision - Volume Part I, 2006

[5] P. Baldi and S. Brunak, "Bioinformatics: The machine learning approach", Cambridge, Mass: MIT Press, 1998

[6] D.D. Lewis, "Naive (Bayes) at forty: The independence assumption in information retrieval", Proceedings of ECML '98.

[7] J.D.M. Rennie, L. Shih, J. Teevan, D.R. Karger, "Tackling the Poor Assumptions of Naive Bayes Text Classifiers", Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003

[8] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial Naive Bayes for Text Categorization Revisited", AI 2004: Advances in Artificial Intelligence pp 488-499, 2004

[9] L. Jiang, Z. Cai, H. Zhang, D. Wang, "Naive Bayes text classifiers: a locally weighted learning approach", Journal of Experimental & Theoretical Artificial Intelligence, 25:2, 273-286, 2013

[10] A. Anagaw, Y.L. Chang, "A new complement naïve Bayesian approach for biomedical data classification", Journal of Ambient Intelligence and Humanized Computing, Volume 10, Issue 10, pp 3889–3897, 2019

[11] K. Komiya, N. Sato, K. Fujimoto, Y. Kotani, "Negation Naïve Bayes for Categorization of Product Pages on theWeb", Proceedings of Recent Advances in Natural Language Processing, pages 586–59, 2011

[12] V. Keiser, T. G. Dietterich, "Evaluating Online Text Classification Algorithms for Email Prediction in TaskTracer", CEAS 2009 – Sixth Conference on Email and Anti-Spam, July 16-17, 2009

[13] A. Goyal and R. Mehta, "Performance Comparison of Naïve Bayes and J48 Classification Algorithms", International Journal of Applied Engineering Research, ISSN 0973-4562 Vol.7 No.11, 2012

[14] A. Ashari, I. Paryudi, A M. Tjoa, "Performance Comparison between Naïve Bayes, Decision Tree and k-Nearest Neighbor in Searching Alternative Design in an Energy Simulation Tool", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 4, No. 11, 2013

[15] N. Friedman, M. Goldszmidt, "Building Classifiers using Bayesian Networks", AAAI-96 Proceedings, 1996

[16] R. Nithya, Dr. D. Ramyachitra, P. Manikandan, "An Efficient Bayes Classifiers Algorithm on 10-fold Cross Validation for Heart Disease Dataset", International Journal of Computational Intelligence and Informatics, Vol. 5: No. 3, 2015

[17] H. Y. Mussa, D. Marcus, J. B. O. Mitchell, and R. C. Glen, “Verifying the fully “Laplacianised” posterior Naïve Bayesian approach and more”, J Cheminform, 2015; 7: 27, 2015
[18] M. Andrejiova, A. Grincova, “Classification of impact damage on a rubber-textile conveyor belt using Naïve-Bayes methodology”, Wear, Volumes 414–415, 2018
[19] R.O. Duda, P.E. Hart, “Pattern classification and scene analysis”, Wiley and Sons, Inc., 1973
[20] D. Heckerman, “A tutorial on learning with Bayesian networks”, (Technical Report MSR-TR-95-06), Microsoft Research, 1995
[21] K. Lang, “Newsweeder: Learning to filter netnews”, Proceedings of the Twelfth International Conference on Machine Learning, pages 331-339, 1995