Comparative Study Of Various Text Classification Methods

Samarth Khanna, Bishnu Tiwari, Priyanka Das* and Asit Kumar Das

Indian Institute of Engineering Science and Technology, Shibpur,
Howrah - 711103, India
samarthkhanna0@gmail.com, bishnutiwari963@gmail.com,
priyankadas700@gmail.com, akdas@cs.iiests.ac.in

Abstract. With the exponential growth in the enhancement of modes of information exchange, the spread of text has become not only substantially faster, but also widespread. Due to this, text has become an indispensable part of all kinds of decision making. Hence it has become imperative to analyse the methods that can help make sense of this text as efficiently as possible. We shall make an attempt at the same by discussing various tools to make this very task increasingly productive. We shall try to analyse the relationship between the way an algorithm works and how it performs on various sets of data having different types of featurization. We shall analyse featurization techniques such as Bag of words/N-grams, Tf-Idf vectorization, average Word2Vec and Tf-Idf Word2Vec.

Keywords: give five

1 Introduction

The burst in availability of information and with the advent of social media, the ease of access to text data has increased manifold to a point where people in all parts of the world have the opportunity to present their opinion in a myriad of ways, the most important probably being in the written form. For example, the data containing the reviews of the products of a particular e-commerce website (amazon in the case of our study) can help us gain insight into the relationship between the demands and needs of a wide spectrum of consumers with the current products in the market. Such analysis could significantly improve both selection of goods to be produced as well as the way they are brought to the consumers. This requires the recognition of the sentiment of a particular text. Compared to the simple topic-based classification this task is fairly complex as it involves the discrimination based on opinions, feelings and attitudes contained in text. The text that is available as raw data is initially not conducive and also it contains a lot of noise in the form of unnecessary characters, words containing numbers and website links etc. that make it difficult to draw inference from the

* Corresponding author

text in the most optimal way. We shall attempt at removing all these hindrances by applying some basic text preprocessing techniques and selecting the most vital words contributing to our task.
Extensive research has already been conducted in this domain that compares the performance of various algorithms on various types sets of data processed/vectorized in different ways. A text classification survey [1] has been carried out by a group of researchers who compared several text feature extractions, dimensionality reduction approaches and other existing classification algorithms. They have also discussed the disadvantages associated to each technique. Before proceeding to text classification, it is important to remove the presence of noisy characters from the data. [2] proposed a research article on stopwords, filtering and discussed several issues of data sparsity in twitter sentiment classification. Sentiment analysis (based on 2000 documents) has been also done in [3] based on various feature selection methods and the classification performance has been evaluated based on precision, recall and accuracy. K Nearest Neighbour (KNN) is a basic classifier but the value of 'k' is automatically determined and varied for different data in [4]. This results in optimal classification accuracy. Again, KNN also has been used with Term Frequency-Inverse Document Frequency (TF-IDF) approach in [5]. This experiment showed the good and bad features of the algorithm. A text categorization approach based on multinomial Naïve Bayes classifier model was discussed in [6]. Locally weighted learning has been used to overcome the basic disadvantages of the basic Naïve Bayes classifier. One of the most widely used text classifier is Support Vector Machine (SVM) [7] as it classifies the text data efficiently. SVMs are robust classifiers and help in improving the basic classifiers over a variety of different learning tasks. A survey focusing on the characteristics, limitations, efficiency of several algorithms of Decision tree [8] like ID3, C4.5 and CART.
The proposed work focuses on four different kinds of featurization techniques likes Bag of words/N-grams, Tf-Idf vectorization, average Word2Vec and Tf-Idf Word2Vec. On each of the data sets that come out of these feature selection methods we shall be studying the effect of 5 major classification algorithms which are K-Nearest Neighbors, Nave Bayes, Logistic regression, Support Vector Machines (SVM) and Decision Trees. We shall be comparing the effectiveness in each case by plotting the Receiver Operating Characteristic curve and checking the value of the area under the curve (ROC-AUC). For this purpose, we shall be using two data sets. The first is an 'Amazon Fine Food Reviews' data set containing 364,173 unique reviews. We shall be considering 10000 out of these for our analysis. The second data set in the 'DonorsChoose' data set that contains information about the various projects that teachers upload for attracting donations. Along with text data, this data set contains numerical as well as categorical data that we shall also consider during the task of classification. Our main aim is to find out which combination of feature selection along with algorithm works better than the rest.

The rest of the paper is organised as follows: Section 2 briefly discusses the existing featurization techniques, section 3 gives an overview of the basic classi-

fication algorithms. Section 4 demonstrates the experimental results and finally section 5 draws the conclusion.

2 Different Featurization Techniques

This section briefly introduces some of the existing featurization techniques used in several applications of data mining.

2.1 Bag of Words/N-grams

This technique involves creating a sparse vector of all the words that are present in the corpus and incrementing the count of each word occurring in a particular document, in the vector corresponding to the document, as many times as the word occurs. In our application, using the CountVectorizer provided by sklearn.feature_extraction, we are setting the min_df and max_features parameters to 10 and 10000 respectively.

2.2 Term Frequency - Inverse Document Frequency (TF-IDF)

Instead of the frequency of the words being present in the sparse vector corresponding to the document, here, a weighted frequency measure is used. This weighted measure is the product of two values, namely, the 'term frequency' (Tf) and the 'inverse document frequency' (Idf). The 'Tf' simply refers to the number times the word occurs in the concerned document. The 'Idf' refers to the logarithm of the ratio between the total number of words in the corpus and the number of times the word occurs in the corpus. We have used the TfidfVectorizer provided by sklearn.feature_extraction.

2.3 Average Word2Vec

Each word is assigned a vector of reasonable dimensions (300 in our case). These vectors are assigned in manners that words having similar meanings are found proportionately close to one another in the vector space. We have used the pre-existing 'word2vec-GoogleNews-vectors' repository for our purpose. This repository hosts the word2vec pre-trained Google News corpus (3 billion running words) word vector model (3 million 300-dimension English word vectors). To create an 'average word2vec' vector corresponding to each document, we sum up all the vectors of the individual words in the document and divide that by the total number of words in the same.

2.4 Tf-Idf weighted Word2Vec

As compared to 'average word2vec' the only difference in this method of feature extraction is that instead of simply adding the vectors corresponding to each word present in the document, the vectors are first multiplied with the product

of term frequency and inverse document frequency and then added. The sum of the vectors is then divided by the sum of the weights for each word and what results is the final document vector.

3 Different Classification Algorithms

Once the featurization techniques have been applied on the datasets to obtain the optimal set of features, next it is the classification techniques that come into play. This section briefly discusses some existing classifiers used in text categorization problems.

3.1 Hyperparameter Tuning

Making use of the GridSearchCV algorithm for cross-validation provided by sklearn.model_selection, we search for the most suitable hyperparameter out of a list that we initialize ourselves. For this purpose, we have used the metrics of 'roc-auc' score and 'negative mean squared error' to measure performance.

3.2 K Nearest Neighbors Classifier

Given a test document x, the algorithm identifies the k-nearest neighbours to x in the training set and accordingly judges the class label of x. This can be done simply by a majority method that gives x the class label that a greater number of neighbours have. We have used the KNeighboursClassifier provided by sklearn.neighbors.

3.3 Naïve Bayes Classifier

This is a probabilistic classifier based on the principals of Bayes theorem and conditional probability. This algorithm assumes conditional independence amongst all the features. We have used the MultinomialNB classifier provided by sklearn.naive_bayes.

3.4 Logistic Regression

The basic idea of this algorithm is to find a hyperplane that most appropriately separates two classes from one another in the multidimensional vector space. We have made use of the SGDClassifier (with 'loss' parameter set to 'log' for logistic regression) provided by sklearn.linear_model that employs stochastic gradient descent to solve the optimization problem.

3.5 Support Vector Machine

This algorithm is based on the Structural Risk Minimization principle based on computational learning theory. It basically tries to find the hypothesis that guarantees the lowest true error. The key idea of support vector machines is to separate positive and negative points in a data set, as widely as possible.

3.6 Decision Tree Classifier

Decision Trees are easy to interpret and visualize. Features don't need to be scaled for this algorithm to perform well. It can also capture non-linear patterns easily. However, Decision Trees are sensitive to noisy data and can over fit due to it. Small changes in the data can change the structure of the tree. We have made use of the DecisionTreeClassifier provided by sklearn.tree, considering maximum permissible depth and minimum samples to split as hyperparameters.

4 Experimental Results

This section introduces the datasets taken for this comparative study, the preprocessing done before application of the featurization and classification techniques. It also demonstrates the results obtained by this comparative study.

4.1 Data collection

The data considered in the proposed work are 'Amazon Fine Food Reviews' dataset and the 'DonorsChoose' dataset. The data source for 'Amazon Fine Food Reviews' data is https://www.kaggle.com/snap/amazon-fine-food-reviews and the data source for 'DonorsChoose' dataset is https://www.kaggle.com/c/donorschoose-application-screening/data.
The 'Amazon Fine Food Reviews' dataset has a total of 10 attributes, out of which the most useful or relevant ones taken into consideration are: ProductId - unique identifier of the product, HelpfulnessNumerator - number of users who found the review useful, HelpfulnessDenominator - number of users who indicated whether the review was useful or not, Score - rating between 1 – 5, Summary - brief summary of the review, and finally, the text of the review.
The 'DonorsChoose' dataset has an additional file containing the related resourses required for the project corresponding to the datapoints of the train and test datasets. This dataset has both text and numerical data. There is also a vast category of attributes to be considered while handling this dataset. A detailed description of the dataset is given in the link provided above.

4.2 Text Preprocessing

Various preprocessing steps that we have followed are realized with the help of the 'regular expression operations' (re) library of python. First, we have decontracted the words, we are replacing various contracted words (combination of multiple words) with their de-contracted forms. Next, we shall remove URLs from the document. After this, we shall remove words that contain numbers and expressions that are purely numbers. Special character and punctuation marks are also removed. We have also removed the tags present in the document using the BeautifulSoup library. Stop-word removal and de-capitalization is done using a pre-defined list of words that do not contribute to the sentiment of the

document, we shall check the presence of each word that we come across in our corpus, in our list. If the word is absent, we add the de-capitalized version of this word to the modified document which in turn will get added to the modified corpus.

4.3 Results based on Amazon Fine Food Review Dataset

After preprocessing the data, the classifiers have been run on the dataset and the results are given in Table 1 – Table 5.

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
K Nearest Neighbours	Bag of Words	n_neighbours = 25	NA	0.838038	0.757655
K Nearest Neighbours	TF-IDF	n_neighbours=25	NA	0.838038	0.757655
K Nearest Neighbours	Average Word2Vec	n_neighbours=50	NA	**0.901414**	**0.863849**
K Nearest Neighbours	TF-IDF Word2Vec	n_neighbours=50	NA	0.891285	0.851164

Table 1: Result of K Nearest Neighbour classifier

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Naïve Bayes	Bag of Words	alpha = 0.1	NA	**0.997334**	0.881323
Naïve Bayes	TF-IDF	alpha = 0.1	NA	0.838038	0.757655
Naïve Bayes	Average Word2Vec	NA	NA	NA	NA
Naïve Bayes	TF-IDF Word2Vec	NA	NA	NA	NA

Table 2: Result of Naïve Bayes classifier

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Logistic Regression	Bag of Words	alpha = 0.001	NA	0.999024	0.925878
Logistic Regression	TF-IDF	alpha=0.0001	NA	**0.999560**	**0.946232**
Logistic Regression	Average Word2Vec	alpha=0.0001	NA	0.976259	0.922959
Logistic Regression	TF-IDF Word2Vec	alpha=0.001	NA	0.947135	0.900259

Table 3: Result of Logistic Regression classifier

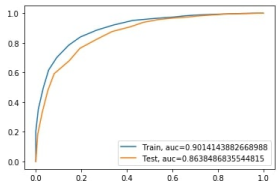
In order to evaluate the effectiveness of the various featurization techniques and algorithms used in this study, the area under the ROC curve is used. Fig. 1 shows the ROC curves for all the classifiers taken into consideration.

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Support Vector Machine	Bag of Words	alpha = 0.001	NA	0.988139	0.753992
Support Vector Machine	TF-IDF	alpha = 0.0001	NA	**0.998986**	**0.759983**
Support Vector Machine	Average Word2Vec	alpha = 0.001	NA	0.760989	0.714596
Support Vector Machine	TF-IDF Word2Vec	alpha = 0.001	NA	0.741939	0.684694

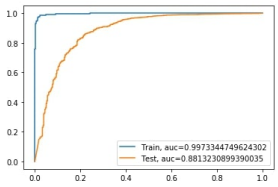
Table 4: Result of Support Vector Machine classifier

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Decision Tree	Bag of Words	max_depth=50	min_sample_split=100	**0.793066**	**0.660423**
Decision Tree	TF-IDF	max_depth=10	min_sample_split=100	0.686137	0.641424
Decision Tree	Average Word2Vec	max_depth=50	min_sample_split=100	0.766898	0.629769
Decision Tree	TF-IDF Word2Vec	max_depth=5	min_sample_split=50	0.650196	0.611293

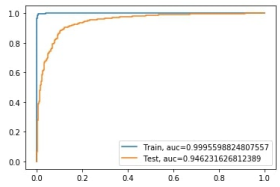
Table 5: Result for Decision Tree classifier



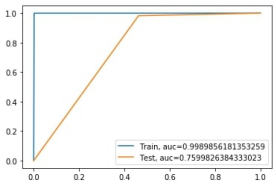
(a) KNN with Average Word2Vec



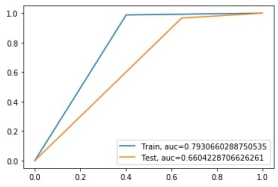
(b) Naïve Bayes with Bag of Words



(c) Logistic Regression with TF-IDF



(d) SVM with TF-IDF



(e) Decision Tree with Bag of Words

Fig. 1: Receiver Operating Characteristics Curve for Amazon Fine Food Review Dataset

It is observed that the K Nearest Neighbours algorithm has proven most effective on the dataset obtained after the Average Word2Vec featurization, as it has the highest ROC-AUC value. The algorithm works well on both the training and test set, indicating its good performance. Naïve Bayes algorithm performs better with the Bag of Words featurization. The performance is good on both the training and test set, indicating that the combination works well for the dataset.

We also observe that Logistic Regression performs good with every featurization technique. The algorithm performs only marginally better in case of the TF-IDF vector. So, we can conclude that Logistic Regression works well with every featurization technique, and is best when used with TF-IDF featurization. We find that the Support Vector Machine and TF-IDF gives the best results. The Decision Tree algorithm has been analysed, and it is found to be best suited for use with the Bag of Words featurization.

4.4 Results based on DonorsChoose Dataset

The classifiers have been separately run on the DonorsChoose dataset and the results obtained are given in Table 6 – Table 10.

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
K Nearest Neighbours	Bag of Words	n_neighbours = 75	NA	1.0	0.580634
K Nearest Neighbours	TF-IDF	n_neighbours = 35	NA	1.0	0.568683
K Nearest Neighbours	Average Word2Vec	n_neighbours = 50	NA	0.648431	**0.553368**
K Nearest Neighbours	TF-IDF Word2Vec	n_neighbours = 35	NA	**0.665354**	0.536918

Table 6: Result of K Nearest Neighbour classifier

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Naïve Bayes	Bag of Words	alpha = 0.001	NA	**0.973727**	0.601970
Naïve Bayes	TF-IDF	alpha = 0.1	NA	0.961296	**0.613406**
Naïve Bayes	Average Word2Vec	NA	NA	NA	NA
Naïve Bayes	TF-IDF Word2Vec	NA	NA	NA	NA

Table 7: Result of Naïve Bayes classifier

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Logistic Regression	Bag of Words	alpha = 0.1	NA	**0.977472**	0.634283
Logistic Regression	TF-IDF	alpha = 0.0001	NA	0.957533	0.608550
Logistic Regression	Average Word2Vec	alpha = 0.0001	NA	0.724182	0.632226
Logistic Regression	TF-IDF Word2Vec	alpha = 0.0001	NA	0.746343	**0.637386**

Table 8: Result of Logistic Regression classifier

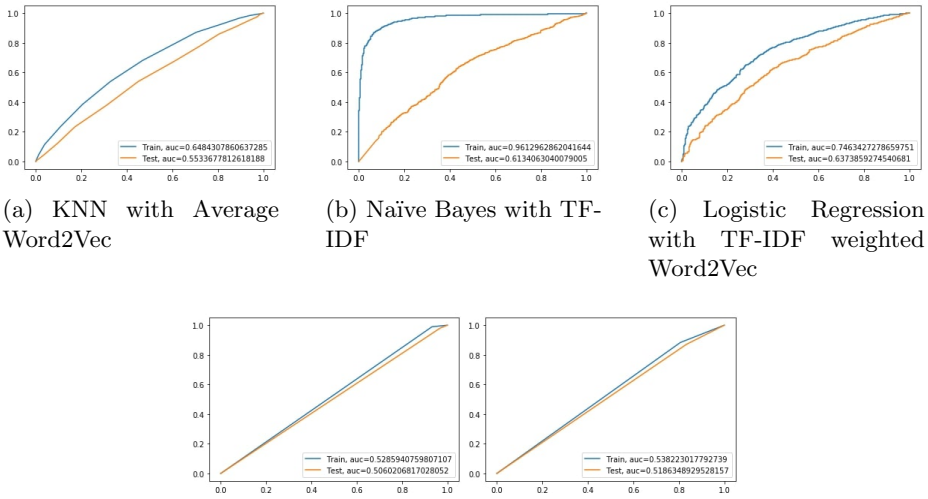
Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Support Vector Machine	Bag of Words	alpha = 0.1	NA	0.5	0.5
Support Vector Machine	TF-IDF	alpha = 0.001	NA	0.5	0.5
Support Vector Machine	Average Word2Vec	alpha = 0.00001	NA	**0.528594**	**0.506020**
Support Vector Machine	TF-IDF Word2Vec	alpha = 0.01	NA	0.5	0.5

Table 9: Result of Support Vector Machine classifier

Model	Featurization	Hyperparameter 1	Hyperparameter 2	Train AUC	Test AUC
Decision Tree	Bag of Words	max_depth = 10	min_sample_split = 50	0.583667	0.507400
Decision Tree	TF-IDF	max_depth = 10	min_sample_split = 5	**0.614716**	0.505951
Decision Tree	Average Word2Vec	max_depth = 5	min_sample_split = 100	0.537034	0.506989
Decision Tree	TF-IDF Word2Vec	max_depth = 5	min_sample_split = 30	0.538223	**0.518635**

Table 10: Result for Decision Tree classifier

Here also, we have evaluated the classifiers based on ROC curves as shown in Fig. 2.



(a) KNN with Average Word2Vec

(b) Naïve Bayes with TF-IDF

(c) Logistic Regression with TF-IDF weighted Word2Vec

(d) SVM with Average Word2Vec

(e) Decision Tree with TF-IDF weighted Word2Vec

Fig. 2: Receiver Operating Characteristics Curve for DonorsChoose Dataset

We find that K Nearest Neighbours algorithm performs exceptionally well on the training set, even giving an AUC of 100% during training using the Bag of Words and TF-IDF featurization. But, the highest test set AUC score is approximately 55%, obtained from the Average Word2Vec featurization. Naïve Bayes algorithm works best with TF-IDF techniques. Logistic Regression works remarkably well with TF-IDF weighted Word2Vec. SVM provides good result with Average Word2Vec and Decision Tree works well for TF-IDF weighted Word2Vec.

5 Conclusion

Having collated the results of the entire experiment, we observe that the most effective algorithm for both the data sets considered is Logistic Regression. In the case of the 'Amazon Fine Food Reviews' data set, the best result is observed with the Tf-Idf featurization. In the case of the 'DonorsChoose' data set, the best result observed with the Tf-Idf weighted Word2Vec embedding of the 'essay' feature. Researchers can refer to this study when applying the discussed algorithms on similar data sets containing text features. However the performance is not satisfactory and there is a need to explore newer, modern algorithms. There is a scope for the development of even newer algorithms that solve the problem a hand to a greater extent, by overcoming the limitations of the algorithms explored.

References

1. Kowsari, K., Meimandi, K.J., Heidarysafa, M., Mendu, S., Barnes, L.E., Brown, D.E.: Text classification algorithms: A survey. CoRR **abs/1904.08067** (2019)
2. Saif, H., Fernandez, M., He, Y., Alani, H.: On stopwords, filtering and data sparsity for sentiment analysis of twitter. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14). (2014) 810–817
3. Sharma, A., Dey, S.: Performance investigation of feature selection methods and sentiment lexicons for sentiment analysis. IJCA Special Issue on Advanced Computing and Communication Technologies for HPC Applications **ACCTHPCA**(3) (July 2012) 15–20
4. Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K.: Knn model-based approach in classification. In: On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE. (2003) 986–996
5. Trstenjak, B., Mikac, S., Donko, D.: Knn with tf-idf based framework for text categorization. Procedia Engineering **69** (2014) 1356 – 1364
6. Kibriya, A.M., Frank, E., Pfahringer, B., Holmes, G.: Multinomial naive bayes for text categorization revisited. In: AI 2004: Advances in Artificial Intelligence. (2005) 488–499
7. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Machine Learning: ECML-98. (1998) 137–142
8. Himani Sharma, S.K.: A survey on decision tree algorithms of classification in data mining. International Journal of Science and Research **5**(4) (2016) 2094 – 2097