Document Classification Using KNN with Fuzzy Bags of Word Representation

P.Lakshmi Prasanna, S.Manogni , P.Tejaswini , K.Tanmay Kumar , K.Manasa

Abstract— Text classification is used to classify the documents depending on the words, phrases and word combinations according to the declared syntaxes. There are many applications that are using text classification such as artificial intelligence, to maintain the data according to the category and in many other. Some keywords which are called topics are selected to classify the given document. Using these Topics the main idea of the document can be identified . Selecting the Topics is an important task to classify the document according to the category. In this proposed system keywords are extracted from documents using TF-IDF and Word Net. TF-IDF algorithm is mainly used to select the important words by which document can be classified. Word Net is mainly used to find similarity between these candidate words. The words which are having the maximum similarity are considered as Topics(keywords). In this experiment we used TF-IDF model to find the similar words so that to classify the document . Decision tree algorithm gives the better accuracy for text classification when compared to other algorithms fuzzy system to classify text written in natural language according to topic. It is necessary to use a fuzzy classifier for this task, due to the fact that a given text can cover several topics with different degrees. In this context, traditional classifiers are inappropriate, as they attempt to sort each text in a single class in a winner-takes-all fashion. The classifier we propose automatically learns its fuzzy rules from training examples. We have applied it to classify news articles, and the results we obtained are promising. The dimensionality of a vector is very important in text classification. We can decrease this dimensionality by using clustering based on fuzzy logic. Depending on the similarity we can classify the document and thus they can be formed into clusters according to their Topics. After formation of clusters one can easily access the documents and save the documents very easily. In this we can find the similarity and summarize the words called Topics which can be used to classify the Documents.

INTRODUCTION

Document classification is very important now a days as social media data and the other data are increasing day by day. So, to classify all the documents and all the data there are different models like RNN,CNN,TOPIC MODELING etc., but now in this we are using KNN with FUZZY BOW(bag of words)[5].KNN is used to find the minimum distance from the query instance to the training samples to determine the K-nearest neighbours, the data for KNN algorithm consist of several multivariate attributes name that will be used to classify[7]. In Bag of words model we use TF-idf and WORD NET. TF-IDF is used to find the important words which are used to classify the document, which is a very important task in classification. WORD NET acts as a database that it is used in finding the similarity between the candidate words.KNN is more effective for large amount of data. So, in this model we considered large number of documents for classification. To use KNN more effectively we use fuzzy with knn as it can promote the precision and recall of text categorization to a certain degree.

RELATED WORK:

The proposed model is fuzzy bag of words model which can be very useful in classification by reducing the extra effort. using fuzzy bag of words the classified words are stored in the bag which will be used in classification. There are many measures to implement fuzzy bag of words model. This should be free from sparsity, high dimensionality, and inability of capturing the semantic meanings of the text. To make this semantic matching of words are replaced the word to word exact matching by semantic matching as this is more prominent . The fuzzy bag of words could encode more semantics into the numerical representation. In this we mentioned about K-Nearest neighbor as this has it's own importance in forming the clusters, so by using KNN with fuzzy bag of words can give the best results with in the limit.TF-idf is used to generate the candidate keys and those are used by knn with fuzzy bag of words to categorise the document accordingly to acquire highest classification accuracies[1].

PROPOSED MODEL:

In this paper we used KNN classifier, TF-IDF, WORDNET, Fuzzy Bag of Words.

KNN (K-NEAREST NEIGHBOUR): Here we mainly talk about the widely used machine learning classification technology called KNN. It stores all available cases and classifies similar cases based on similarity [10], which is unsupervised learning.

Revised Version Manuscript Received on March 08, 2019.

P.Lakshmi Prasanna, Assistant Professor , KL university, Guntur, Andhra Pradesh, India

S.Manogni, B.Tech graduate, KL university, Guntur, Andhra Pradesh, India

P.Tejaswini, B.Tech graduate, KL university, Guntur, Andhra Pradesh, India

K.Tanmay Kumar, B.Tech graduate, KL university, Guntur, Andhra Pradesh, India

K.Manasa, B.Tech graduate, KL university, Guntur, Andhra Pradesh, India

*Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication*

It is usually used to search for applications where you are looking for the same type of project. It can be used for regression and classification problems. The prediction of test data is based on its behavior. K is an integer, if (k = 1), K is assigned to the class of a single nearest neighbor [11].

Example: consider a data to classify whether it is good or bad.

A=Base Classifier	B=Strength	X=Classification
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Determine the parameter K=number of nearest neighbour, Suppose K=3, Query instance (3,7).

A=Base Classifier	B=Strength	distance
7	7	4
7	4	5
3	4	3
1	4	3.6

Sort the distance and determine the minimum.

A=Base Classifier	B=Strength	Distance	Rank minimum distance	3 as a nearest neighbour or not
7	7	4	3	Yes
7	4	5	4	No
3	4	3	1	Yes
1	4	3.6	2	Yes

Merge the category of x=Classification, nearest neighbour in the second row last column is not included because rank of data is more than 3.

A=Base Classifier	B=Strength	Distance	Rank minimum distance	3 as a nearest neighbour or not	X=Classifcation
7	7	4	3	yes	Bad
7	4	5	4	no	-
3	4	3	1	yes	Good
1	4	3.6	2	Yes	Good

Here by seeing the majority simply we can say that the classification is good.

TF-IDF:

Term and inverse document frequency which is used in text mining and information retrieval. This is used by many search engines for ranking the document by the user. In a document how many times a word repeated or appear its tf-idf value increases [1]. Here the weight of the term is simply proportional to the term frequency. idf provides simply how much information does the word provides. By using tf-idf we can convert the unstructured text to useful features.

WORD NET:

Word Net is a lexical data base for English, it groups the english words into synonyms. This is used to check similarity in the words of the candidate. It is primarily used in text analysis and artificial intelligence. The main goal of word net is to construct a lexical data base with the theories of human semantic memory, it has successfully applied in many human languages. One major shortage is poor expressive capability, due to cost of hand-coding, but the synonyms provide the possibility to generate lexical paraphrases.

FUZZY BAG OF WORDS:

FBOW is also known as vector space model. Here a sentence is represented as a multiset of words without any priority of grammar. It is also used for computer vision. Each element here represents a number based on the frequency of the term, bag of words is precisely matched but the semantic meaning behind the data is captured due to extreme sparsity and high dimension. FBOW is used for mainly document representation, image classification.

Example:

D1:” I am feeling good today”.

D2:” I am going to movie today”.

Based on the two documents it creates a vocabulary using same words “I am feeling going good to movie today”.

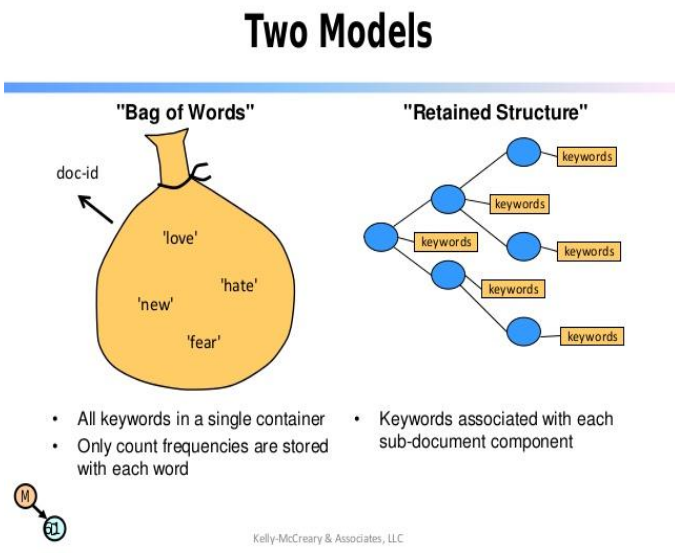
For each word the frequency is inserted.

	I	Am	Feeling	Going	Good	To	Movie	today
D1	1	1	1	0	1	0	0	1
D2	1	1	0	1	0	1	1	1

Thus the above table indicates the term frequencies of each word in a document.

ARCHITECTURE :

DOCUMENT REPRESENTATION FOR FUZZY BAG OF WORDS



Two Models

"Bag of Words"

"Retained Structure"

- All keywords in a single container
- Only count frequencies are stored with each word

- Keywords associated with each sub-document component

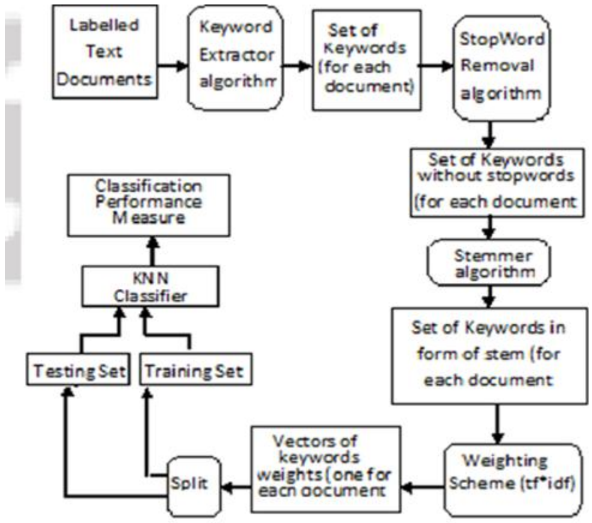

Labelled Text Documents
Keyword Extractor algorithm
Set of Keywords (for each document)
StopWord Removal algorithm
Set of Keywords without stopwords (for each document
Stemmer algorithm
Set of Keywords in form of stem (for each document
Weighting Scheme (tf*idf)
Vectors of keywords weights (one for each document
Split
Testing Set
Training Set
KNN Classifier
Classification Performance Measure

Figure 1: Flow of text classification

ALGORITHM:

Here we are using Bag of words and KNN for document classification,

Algorithm for Bag of Words: It is used in order to perform the term frequency of a document.[1]

Bag of Words:

1. Bag of words ignores grammar and arrangement of words.

2. Here we start with two documents which is known as corpus.

3. A list is created based on the unique words in the corpus.

3.1 Here we will use count Vectorizer to create vectors from the corpus.

3.2 It counts the term frequency based on the documents.

4. Thus Bag of Words is implemented.

KNN:

1. First, we have to use the data for loading

2. Initialise the K value

3. For getting the anticipated class, emphasize from 1to all out information focuses.

3.1 compute the separation between the test information and the train information.

3.2 We use Euclidean distance formula as a metric.

3.3 Based on the distance values arrange them in the ascending order.

3.4 From the sorted array see the top k values.

3.5 See the most frequent class.

3.6 Return the predicted class.

Thus KNN will work.[10,11,12]

TF-IDF :

Step1:consider the texts which you want to classify.

Step2: calculate the term frequency for each term in text

Step3:calculate the inverse document frequency value for each document

Idf = log(N/dft)

Here N=number of words in text

Step4: compute the term frequency and inverse document frequency

Tf-idf :term frequency * Idf

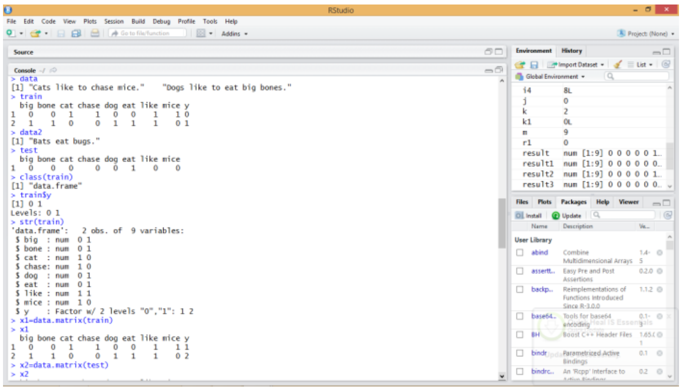
Ex: Text1 – Classification is used for mining

Text2 – Classification is not used for mining

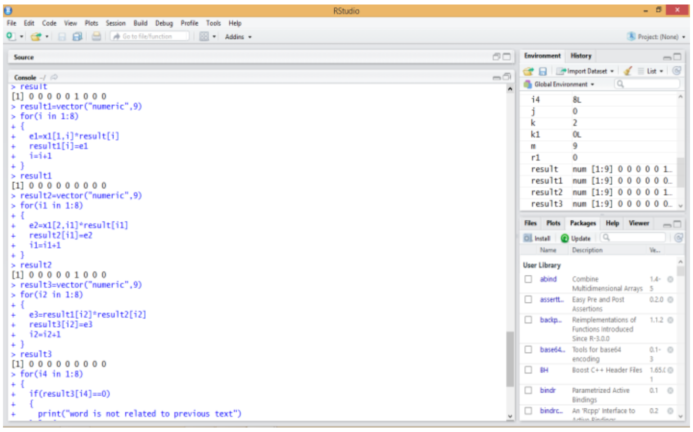
Words/query	A	B	IDF	TfIdf(A)	Tfidf(B)	Tfidf(A) * Tfidf(B)
Classification	1	1	Log(2/2)=0	0	0	0
Is	1	1	Log(2/2)=0	0	0	0
Used	1	1	Log(2/2)=0	0	0	0
For	1	1	Log(2/2)=0	0	0	0
Mining	1	1	Log(2/2)=0	0	0	0
Not	0	1	Log(1/2)=1	0	1	0

In the last column the value 0 indicates that that word is not related to the particulare document

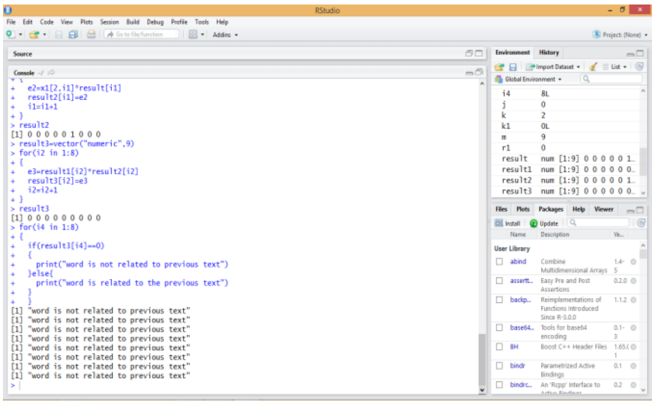
RESULTS :


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
Console ~/
Environment History
Files Plots Packages Help Viewer

```
> data
[1] "Cats like to chase mice."    "Dogs like to eat big bones."
> train
  big bone cat chase dog eat like mice y
1   0    0   1     1   0   0    1    1 0
2   1    1   0     0   1   1    1    0 1
> data2
[1] "Bats eat bugs."
> test
  big bone cat chase dog eat like mice
1   0    0   0     0   0   1    0    0
> class(train)
[1] "data.frame"
> train$y
[1] 0 1
Levels: 0 1
> str(train)
'data.frame':   2 obs. of  9 variables:
 $ big  : num  0 1
 $ bone : num  0 1
 $ cat  : num  1 0
 $ chase: num  1 0
 $ dog  : num  0 1
 $ eat  : num  0 1
 $ like : num  1 1
 $ mice : num  1 0
 $ y    : Factor w/ 2 levels "0","1": 1 2
> x1=data.matrix(train)
> x1
  big bone cat chase dog eat like mice y
1   0    0   1     1   0   0    1    1 1
2   1    1   0     0   1   1    1    0 2
> x2=data.matrix(test)
> x2
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
Console ~/
Environment History
Files Plots Packages Help Viewer

```
> result
[1] 0 0 0 0 0 1 0 0 0
> result1=vector("numeric",9)
> for(i in 1:8)
+ {
+   e1=x1[1,i]*result[i]
+   result1[i]=e1
+   i=i+1
+ }
> result1
[1] 0 0 0 0 0 0 0 0 0
> result2=vector("numeric",9)
> for(i1 in 1:8)
+ {
+   e2=x1[2,i1]*result[i1]
+   result2[i1]=e2
+   i1=i1+1
+ }
> result2
[1] 0 0 0 0 0 1 0 0 0
> result3=vector("numeric",9)
> for(i2 in 1:8)
+ {
+   e3=result1[i2]*result2[i2]
+   result3[i2]=e3
+   i2=i2+1
+ }
> result3
[1] 0 0 0 0 0 0 0 0 0
> for(i4 in 1:8)
+ {
+   if(result3[i4]==0)
+   {
+     print("word is not related to previous text")
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
Console ~/
Environment History
Files Plots Packages Help Viewer

```
+   e2=x1[2,i1]*result[i1]
+   result2[i1]=e2
+   i1=i1+1
+ }
> result2
[1] 0 0 0 0 0 1 0 0 0
> result3=vector("numeric",9)
> for(i2 in 1:8)
+ {
+   e3=result1[i2]*result2[i2]
+   result3[i2]=e3
+   i2=i2+1
+ }
> result3
[1] 0 0 0 0 0 0 0 0 0
> for(i4 in 1:8)
+ {
+   if(result3[i4]==0)
+   {
+     print("word is not related to previous text")
+   }else{
+     print("word is related to the previous text")
+   }
+   }
[1] "word is not related to previous text"
[1] "word is not related to previous text"
[1] "word is not related to previous text"
[1] "word is not related to previous text"
[1] "word is not related to previous text"
[1] "word is not related to previous text"
[1] "word is not related to previous text"
[1] "word is not related to previous text"
> 
```

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

CONCLUSION :

From the above results we can conclude that the text classification is done through similarity index values .In the tf-idf algorithm we use the similarity index values in the vector so that the values in the vectors are multiplied with the other set of values in the other set of vectors .The output is describing whether the word in each line is present in the document or not with similarity between the words in the text as well as the queried line. From the the output each word similarity index values or also displayed in this value .In this way we can classify whether the queried text is similar to the text which is already in the database of R programming language.

REFERENCES:

1. Fuzzy Bag-of-Words Model for Document Representation (Base Paper) by Rui Zhao and Kezhi Mao.
2. Classification Algorithms for Data Mining: A Survey by Raj Kumar and Dr.RajeshVerma
3. An Efficient Classification Approach for Data Mining by Hem Jyotsana Parashar, Singh Vijendra, and Nisha Vasudeva
4. Similarity-based Classification: Concepts and Algorithms by Yihua Chen, Eric K. Garcia, Maya R. Gupta
5. An Optimized K-Nearest Neighbor Algorithm for Large Scale Hierarchical Text Classification by Xiaogang Han, Junfa Liu, Zhiqi Shen and Chunyan Miao
6. Robust Kernel Density Estimation by JooSeuk Kim and Clayton D. Scott
7. An Improved k-Nearest Neighbor Classification Using Genetic Algorithm by N. Suguna , and Dr. K. Thanushkodi
8. The Research of Data Mining Classification Algorithm that Based on SJEP by Liang Zhao , Deng-Feng Chen , Sheng-Jun Xu and Jun Lu
9. Offensive Decoy Technology for Cloud Data Attacks by Lingaswami, Avinash Reddy
10. An Optimized K-Nearest Neighbor Algorithm for Large Scale Hierarchical Text classification
11. Robust Kernel Density Estimation by JooSeuk Kim .
12. An Improved k-Nearest Neighbor Classification Using Genetic Algorithm.
13. Classification of Indian Stock Market Data Using Machine Learning Algorithms.
14. Fuzzy Approach Topic Discovery in Health and Medical Corpora.
15. Fuzzy Clustering for Topic Analysis and Summarization of Document Collections.
16. Bag of Discriminative words representation via topic modeling.
17. Analysis of Initialization method on fuzzy c-means algorithm based on singular value decomposition for topic detection.
18. A study on topic identification using k-means clustering algorithm.
19. Bag of words representation for biomedical time series analysis by jiu wang.
20. Weighted fuzzy rule based sentiment prediction analysis on tweets.
21. using tfidf to determine word relevance in document queries by juan ramos
22. quantification of portrayal concepts using tf idf weighting.
23. An improvement of TFIDF weighting in text categorization.
24. A novel TFIDF weighting schema for effective ranking.
25. use of TF-IDF to examine the relevance of words of documents