A General Algorithm of Association Rule-Based Machine Learning Dedicated for Text Classification

Zeyad hamid$^{1,2}$ and Hussein K Khafaji$^3$
$^1$ College of Information Technology, University OF Babylon, Iraq
$^2$ Ministry of Iraqi defence, Iraq
$^3$ Computer Communication Engineering Dept, Al-Rafidain University College, Iraq
$^{1,2}$Zeyadhamid2@gmial.com, $^3$dr_hkml811@yahoo.com

Abstract. Many data mining techniques and machine learning algorithms have been developed to classify textual data involving decision tree, support vector machine, K-Nearest neighbour, in addition to machine learning-based algorithms. Association rules based machine learning is accomplished in two phases; training phase and testing phase that may be reinforced to enhance the classification accuracy according to new minimum support and confidence. Association rules mining/processing, in its various applications, passes through two massive computation steps; frequent itemsets mining and association rules extraction. This paper presents a general algorithm for association rules-based machine learning dedicated to text classification. To verify the efficiency of the algorithm, different text datasets were used such as tweets dataset for sentiment classification, pdf documents and HTML documents. Experiments of sentiment classification showed that the classifier constructed according to minsup threshold =%700 and minconf threshold =50% gives the best performance with $F1 = 0.9861811$ while the experiments of HTML and PDF appeared accurate classification equal to (94%).

1. Introduction
Since the unexpected explosion of textual data sharing by peoples, organizations through social media and websites platforms, it is necessary to develop fast and efficient methods for extracting useful features that utilizing in text classification. Machine learning and data mining are two integrated approaches used by many researchers for achieving this goal. Machine learning algorithms are divided into two famous methods; supervised and unsupervised algorithms. Supervised machine learning algorithms are mainly used for data classification while unsupervised machine learning algorithms used for clustering. Text classification is one of the most important applications of supervised machine learning [1]. A classifier based on association rules mining involves two steps; discovering frequent wordsets and mining of association rules for constructing text classifier. The process of finding all frequent wordsets is a very challenging task because the search space is exponential in the number of features that existing in the database. Therefore, many algorithms have been developed to reduce the time and effort required for mining frequent word sets.[2]. In this paper, we introduced an algorithm for finding a frequent wordsets and building an associative classifier for textual data.
2. Related work

Many techniques have been investigated for classifying English documents. Joachims T. (1998) discussed exploiting the support vector machine (SVM) for text classification. He proved theoretical and empirical facts that (SVM) is appropriate for text classification as well as superior other existing techniques. Also, the experiments showed that (SVM) is effortless because it doesn’t need parameters setting and can extract parameters automatically [3]. Jingnian Chen et al (2009) examined the influence of preprocessing and features selection on the accuracy, scalability, and efficiency of the text classifier. They analyzed Naïve Bayes classifier results by using two features evaluation metrics which are Multi-Class Ratio (MCR) and class discriminating measure (CDM). Experiments results showed the proposed methods overcome other features selection methods [4]. Vishwanath Bijalwan et al. (2014) used k-Nearest Neighbors (KNN) for classifying documents relying on machine learning. Comparing with Naïve Bayes, (KNN) yielded more precise results but it suffered from time complexity [5]. Arlina D’unha et al. (2015) suggested a method for classifying scientific papers by adjusted TF-IDF formula. The proposed method depended on the paper structure and features contained in it, additionally, TF-IDF formula used term position weight to get accurate outcomes using Naïve Bayes algorithm [6]. Kewen Chen et al. (2016) proposed TFIGM as an alternative for TF-IDF technique. TF-IGM used a statistical model in order to accurately detect term class and its performance is proven through conducting experiments of using SVM and KNN classifiers [7]. Lucas Dixon et al. (2018) discussed a method for determining and reducing bias in the text classification process, in this research, the authors illustrated resulted bias can introduce from the imbalanced training dataset. Unsupervised machine learning approach has been used to diminish undesired bias based on balancing training datasets. The experiments results proved the efficiency of the proposed method[8]. Irony discovery research conducted on tweeter dataset by Cynthia Van Hee et. al. (2018) to specify whether the tweet ironic or not (task1), then determine the type of ironic tweets (task2). A dataset consists of (3834) training tweets are provided for both tasks. The first task is the binary classification mission whereas the second task is multiclass classification approach. Forty-three teams participated to accomplish the first task and the best-obtained result was (0.705) which is produced by a system developed by THU NGN. On other hand, thirty-one team share in task2 and best F1 score was achieved was (0.507) by UCDCC team[9]. Xilun Chen et. al. (2018) presented the Adversarial Deep Averaging Network (ADAN) algorithm to process the sentiments classification problem in low-resources languages that don’t have enough annotated data. The proposed algorithm used the knowledge acquired from hi-resources language to low-resources languages. Experiments have been done on Chinese and Arabic languages sentiment classification shows superior the proposed approach on state-of-the-art systems [10]. Hadi et al. (2018) proposed a hybrid association classification algorithm called (HAC) to decrease rules number and to generate many rules that represent every attribute value through using Naïve Bayes algorithm. (HAC) produces rules with only one attribute value in their body as well as not perform sorting and pruning methods. Experiments performed on Arabic textual data and standard Reuters-21578 dataset and proposed algorithm appeared better results comparing with J48, NB, classification based on associations (CBA), multiclass classification based on association rules (MCAR), expert multiclass classification based on association rules (EMCAR), and fast associative classification algorithm (FACA) [11]. Wanwan Zheng et al. (2019) conducted a comparative study to pick a feature that uses for text classification. At first, authors list the best thirteen feature selection techniques below five various categories in order to specify their ability and performance. The top five methods were selected for performing multi-class classification. Support vector machine was used on different language and different numbers of selected features. The experiment results show Mahalanobis distance is the best technique [12].

3. Text classification

Most of the textual data is already reachable through Websites and social media networks, in massive collections of text documents kept in an unorganized text documents forms, which produces a problem for a human to reach the required information quickly and accurately. Therefore, it is necessary to preprocessing these collections of documents into a more suitable form and then classified into subject
classes to make them easier for potential users to specify [13]. More precisely, text mining is related with developing and executing some techniques for generating interesting and beneficial features from unstructured text documents. One of the most important text mining tasks is text classification. With the daily increasing number of various collections of documents that available through websites and social media networks, there is an urgent necessity to construct automatic methods to arrange these documents into their respective classes in order to facilitate finding and capturing relevant text [14]. Therefore, automatic text classification is the task of classifying an unorganized text document in its required classes according to its features.

Text pre-processing is considered as the most important stage in the text classification process to reduce the number of features that use to construct the classifier. It involves tokenization, stop word removing and stemming. Tokenization is the process of partitions the text documents into discrete words. While eliminating the useful words from text is the removing stop words process. Stemming is the latest step in the text pre-processing which means removing prefixes and suffixes from the word and returns them to their roots. Preprocessing process is very important for reducing the dimensionality of textual data. Many approaches have been developed for text classification like support vector machine, decision tree, Naïve Bayesian and text classification based association rules.

There are two main steps to construct textual associative classifier are finding frequent itemsets and association rules mining. The first step involves extracting all wordsets that satisfy minimum support determined by the user. Minimum support of an itemset is the number of documents containing the itemsets to the total number of documents. Determining all frequent itemsets is considered as the most difficult and consuming time because the number of frequent itemsets is exponential to the number of frequent items in the database.

4. Association rules mining
Association rules mining was invented by Agrawal et al. in 1993 who produced the Apriori algorithm for exploring and finding relations among items in the database [15]. Association rule mining is one of the important data mining tasks used to uncover hidden relationships among itemsets of the database. This implies existing some items consider a proof of present other items with a specific probability. Confidence and support are two measures that are used to determine the power of discovered rules. Support of rules is the proportion of the transaction that contains items, whereas confidence is the parameter measures the probability of existing specific items in case of present other items in the same transactions [16].

The formal definition of association rules: Let I = \{i1, i2, i3, i4, . . . im\} be a set of items. Let D be a set of transactions, where each transaction t is a set of items such that t \subseteq I. Each transaction has a unique identifier, called TID. A transaction t contains X, a set of some items in I, if X \subseteq t. An association rule is an implication of the form X \Rightarrow Y, where X \subseteq I, Y \subseteq I, and X \cap Y = \emptyset. The rule X \Rightarrow Y holds in D with confidence C if the segment of transactions that also contain Y in those which contain X in D is C. The rule X \Rightarrow Y has support S in D if the segment of transactions in D that contains X \cup Y is S. Given a set of transactions D, the aim of mining association rules is to generate all association rules, which have support and confidence equal and larger than the minimum support (minsup) and minimum confidence (minconf), which are defined by the user [17].

5. The proposed system
In this research, a general machine learning algorithms based on association rules is presented as shown in figure 1. The proposed algorithms are dedicated to text classification. These algorithms are for training and testing phases. Datasets should be pre-processed before the learning process. The training phase also involves frequent itemsets mining in training dataset and classification rules extraction to construct a classifier. While in the testing phase we performed experiments to measure the accuracy of the classification process which results from the training phase. Testing phase roughly has the same stages of the training phase.
5.1. Pre-processing stage

In order to get accurate classification results, we need to conduct pre-processing steps which including tokenization, stop word removing and stemming, in addition to document-format based pre-processing to obtain the text from a predefined format such as pdf, ppt, doc, etc. Tokenization partitions the text into discrete word. Then the ineffective words in the classification process are removed through performing the stop word removing step. Stemming is the last step in the pre-processing stage which means removing prefixes and suffixes from the word and abstracts them to their roots. The pre-processing stage is implemented according to the algorithms presented in [18, 19]. This paper is concentrating on training and testing phase algorithms.

Figure 1. The proposed system architecture
5.2. Association rules learning stage

Rules in the proposed system consist of one or more terms, (words), in the left-hand side and class label on the right-hand side as the following formula:

\[(word_1, word, word_n) \text{ confidence, if } \ldots \text{ class label} \quad (1)\]

The power of rules is depending on predefine confidence and support values. The purpose of the association rule learning stage is extracting classification rules. It consists of two main steps frequent itemsets mining and classification rules extracting.

5.3. Proposed algorithm to extracting textual frequent itemsets

In this section, we present Frequent Wordsets Mining Algorithm (FIWA) to extract frequent textual features as illustrated in Figure 2. The input to this algorithm is the preprocessed data obtained from preprocessing step. Recall that this dataset is transactional dataset such that each transaction consists of three attributes as the following schema: Textual_dataset (DocumentID, set of words, class label).

To accomplish this step the label will be merged with the set of document features, (words), to be as the following schema:

Textual_dataset (DocumentID, set of words ∪ {class label})

The minimum support threshold, minsup, should be specified. The number of discovered wordsets is depending on the predefined minimum support value. First of all, the algorithm finds the 1-wordsets by counting occurrences of each word in documents that greater than or equals the predefined minimum support. Then, it calls the Extract_second_wordsets algorithm for generating the second frequent wordsets as shown in Figure 3.
To find frequent k-wordsets we will use extract_K_wordsets algorithm by passing the second frequent wordsets. The later algorithm is recursive algorithm which is called to mine (k+1)-wordsets as shown in Figure 4. The proposed algorithms scan the dataset once to mine frequent 1-wordsets. The wordsets are stored as a pair (wordset×DocumentID List), i.e., the wordset and list of the documents identifiers containing the word. The generation of the next level of frequent wordsets will be done by two simple operations; the union of two wordsets and intersection of their DIDLists. The union operation united two n-wordsets that have (n-1) common words. The generated (n+1)-wordset will be ordered by sorting the last members of the two n-wordsets.

```
Extract_second_wordsets Algorithm (F_first_wordsets)
Input
  First_F : frequent 1-wordsets;
Output
  Second_F : frequent 2-wordsets;
{  
  1  Second_F = { };  
  2  length= |First_F| ;  
  3  i=1;  
  4  do while ( i<= |First_F|-1)  
  5   {   
  6    next=i;  
  7   }  
  8    fz= First_F_i ∪ First_F_next;  
  9    TID_fz= TID_First_F_i ∩ TID_First_F_next;  
 10   if (|TID_fz|>= minsup) then Second_F= Second_F ∪ {fz};  
 11   Next= next+1;  
 12   i=i+1;  
 13  }  
 14  Return Second_F;
}
```

```
Algorithm (3) Extract_K_wordsets (K, F_{k-1})
Input
  F_{k-1} : frequent (k-1)-wordsets
Output
  Frequent_K : frequent K-wordsets
{  
  1. Frequent_K={ };  
  2. For each pair of wordsets Fx and Fy belong to Frequent_k-1 such that:  
    Fx={x_{i-1}, x_{k-2}, x_{k-1}} and Fy={x_{i-1}, x_{k-2}, Y_{k-1}}  
  3.  
  4. Fz= Fx ∪ Fy;  
  5. TID_Fz= TID_Fx ∩ TID_Fy;  
  6. if (|TID_Fz|>= minsup) then  
  7.    Frequent_K = Frequent_K ∪ {Fz};  
  8. else remove Fz;  
  9. }  
10. K=K+1;  
11. extract_K_wordsets(K, Frequent_K); //Recursive call  
12. Frequent_K=Frequent_K ∪ Frequent_K-1;  
13. Return Frequent_K;
}
```

Figure 3. Extract_second_wordsets

Figure 4. Extract_K_wordsets algorithm
5.4. Classification rules extracting

The frequent wordsets that are discovered in the previous step will be used to create rules to determine the class of new unclassified text. The antecedent in the left side consists of document features, words, or part of its features, while the consequent of the rule is the class of the document as shown in the following Classification Rules Extracting algorithm, CRE, presented in figure (5). CRE selects frequent k-wordsets such that k>=2 and each selected wordset should contain a class label to easily obtain a classification rule according to formula 1. The interestingness of the rule is computed according to the confidence, support, lift, accuracy, etc., as follows:

\[
co(X \rightarrow \text{Class Label}) = \frac{\text{support}(X \cup \text{Class Label})}{\text{support}(X)}
\]

\[
\text{support} (X \rightarrow \text{Class Lable}) = \frac{(X)}{(\text{Number of documents in training dataset})}
\]

\[
\text{lift} (X \rightarrow \text{Class Lable}) = \frac{\text{co}(X \rightarrow \text{Class Label})}{\text{support(class label)}}
\]

\[
\text{Accuracy} (X \rightarrow \text{Class label}) = \frac{(\text{Number of documents in training Dataset}+1)}{(\text{Number of documents in training Dataset}+\text{Number of classes})}
\]

![Classification Rules Extracting Algorithm](algorithm.png)

6. Experiments and results

In this section, we present experiments results that have been done to analyze and determine the effect of preprocessing steps on discovering frequent wordsets by reducing the volume of textual data. Then, we applied Frequent Wordsets Mining Algorithm to find the number of frequent wordsets using various minimum support thresholds. Lastly, we execute the proposed system to measure the accuracy of
classification process. These experiments have been done on various types of datasets such as tweeter, pdf, html, etc.

6.1. Experimental result using tweeter datasets

To execute our experiments we used tweeter datasets includes millions of tweets obtained from the University of Michigan [20]. This dataset has been preprocessed and each tweet related to one of two classes positive and negative depending on the opinion of the user about a specific topic.

6.1.1. Pre-processing experiments results

Table 1 shows the required time and number of tokens acquired after implementing pre-processing stages on three different datasets.

| Dataset size | Tokenization time (m:s) | Stop words removing time (m:s) | Stemming time (m:s) |
|--------------|-------------------------|------------------------------|---------------------|
|              | No. of tokens           | No. of tokens                | No. of tokens       |
| 300000       | 09:75                   | 00:32                        | 08:27               |
|              | 2,150,580               | 1,152,675                    | 1,037,194           |
| 600000       | 22:10                   | 00:68                        | 20:23               |
|              | 4,384,231               | 2,782,431                    | 2,385,673           |
| 900000       | 42:31                   | 02:54                        | 41:78               |
|              | 5,932,368               | 3,143,476                    | 2,985,154           |

6.1.2. Extracting Frequent wordsets results

After the pre-processing stage we obtained structured data that can be used for generating frequent wordsets. To generate different frequent wordsets we used five minimum support thresholds. The five minimum support values and the number of frequent wordsets illustrated in Table 2. This table explains that when minsup = 10%, then the minimum support of frequent wordsets is (70000), i.e., the minimum number of occurrences of wordset is (70000).

| Minsup | Minimum frequency of wordsets in database |
|--------|------------------------------------------|
| 10%    | 70000                                    |
| 1%     | 7000                                     |
| 0.1%   | 700                                      |
| 0.01%  | 70                                       |
| 0.001% | 7                                        |

After executing Frequent Wordsets Mining Algorithm (FIWA) on the dataset we obtained the results as shown in table 3.

It is very obvious from the frequent wordsets table that the number of frequent wordsets is increased as the minimum support value decreased. The first minsup threshold value is 10% = (70000/700000) which didn’t provide any frequent wordsets, therefore the next value was much smaller, it is 1% = (7000/700000). This case produced only frequent 1-wordsets. This also lead to choose much smaller value again, etc. until using very small value which is 0.001% = (7/700000).
Table 3. Frequent wordsets

| Minsup | 70000 | 7000 | 700  | 70   | 7    |
|--------|-------|------|------|------|-----|
| 1-wordsets | 0     | 52   | 698  | 4103 | 17876|
| 2-wordsets | 0     | 0    | 108  | 7310 | 134763|
| 3-wordsets | 0     | 0    | 23   | 217  | 30659|
| 4-wordsets | 0     | 0    | 13   | 63   | 2589 |
| 5-wordsets | 0     | 0    | 4    | 25   | 1376 |
| 6-wordsets | 0     | 0    | 1    | 9    | 1140 |
| 7-wordsets | 0     | 0    | 0    | 1    | 598  |
| 8-wordsets | 0     | 0    | 0    | 0    | 244  |
| 9-wordsets | 0     | 0    | 0    | 0    | 67   |
| 10-wordsets | 0   | 0    | 0    | 0    | 10   |
| 11-wordsets | 0    | 0    | 0    | 0    | 1    |
| Total    | 0     | 52   | 847  | 11728| 189,393|

6.1.3. Classification results

In order to validate the accuracy of the proposed system, tweets of the testing dataset were used for mining positive and negative tweets using obtained classifier from the training phase. We used (IR) measure which includes recall and precision pointers. Recall for a class is defined as the proportion of correctly classified tweets among all tweets belonging to that class, while precision is the proportion of correctly classified tweets among all tweets that were assigned to the class by the classifier. Confusion matrix used to introduce the above measures. Confusion matrix includes actual and predicted results given by classifier. Figure 6 shows confusion matrix parameters.

![Confusion matrix parameters](image)

Table 4. shows the values of TA, FA, and FR, the parameters of IR measures for each set of rules.

Where:

\[
\text{Recall} = \frac{TA}{TA + F} \quad (6)
\]

\[
\text{Precision} = \frac{TA}{TA + FA} \quad (7)
\]

Table 4. Values of IR measures parameters

| Minsup | Minconf  | TA     | FA     | FR     |
|--------|----------|--------|--------|--------|
| 7000   | 70%      | 12589  | 1407   | 2869   |
|        | 50%      | 33750  | 3000   | 1773   |
|        | 30%      | 47598  | 1976   | 2511   |
|        | 100%     | 3965   | 360    | 117    |
|        | 90%      | 6742   | 700    | 785    |
| 700    | 70%      | 58611  | 1249   | 2507   |
|        | 50%      | 115933 | 2089   | 3040   |
|        | 30%      | 133158 | 3952   | 4751   |
Then we used F1 measure to obtain the actual accuracy of the classifier. The F1 equation as follows

$$F1 = 2 \times \frac{\text{recall} \times \text{precision}}{\text{recall} + \text{precision}}$$

(8)

F1 measure values for each set of rule as shown in table (5). It's obvious that many sets of rules have F1 value near to one, which mean that they are precise rules, but the best result we have obtained when the value of minconf = 50% and minsup= 700%. Therefore we can choose this set as classification rules.

| Minsup | Minconf | Recall   | Precision | F1      |
|--------|---------|----------|-----------|---------|
| 7      | 70%     | 0.8144003| 0.8994712| 0.8548244|
|        | 50%     | 0.9500886| 0.9183673| 0.8725303|
|        | 30%     | 0.9498892| 0.9601403| 0.9120269|
| 700    | 100%    | 0.9713375| 0.9167630| 0.9432615|
|        | 90%     | 0.8957087| 0.9059392| 0.9007949|
|        | 50%     | 0.9500886| 0.9183673| 0.8725303|
|        | 30%     | 0.9498892| 0.9601403| 0.9120269|
| 7000   | 70%     | 0.9589809| 0.9791346| 0.9689529|
|        | 50%     | 0.9744479| 0.9982004| 0.9861811|
|        | 30%     | 0.9668797| 0.9845130| 0.9756166|
| 70     | 100%    | 0.8435607| 0.8212521| 0.8322569|
|        | 90%     | 0.9350986| 0.9061151| 0.9203787|
|        | 70%     | 0.9564812| 0.9794435| 0.9678261|
|        | 50%     | 0.9625519| 0.9689877| 0.9657590|
|        | 30%     | 0.9655497| 0.9711764| 0.9683548|
| 7      | 100%    | 0.8106104| 0.8493019| 0.8295052|
|        | 90%     | 0.8843292| 0.9202951| 0.9019537|
|        | 70%     | 0.9457970| 0.9730112| 0.9592111|
|        | 50%     | 0.9658823| 0.9746733| 0.9702578|

6.2. Experiments results using PDF and HTML documents

In this section, we introduced experiments results obtained from the pre-processing step and classification process using HTML and pdf documents. These documents have four class labels: Web Mining, Machine Learning, Image Processing and Neural Networks.

6.2.1. Pre-processing results

PDF source code is considered the main source for text extraction process. To extract text from PDF documents, the stream of characters must be extracted and converted into accepted words without the noise characters such as source code commands and other non-alphabetic characters. An experiment is
done on the total of (30) PDF documents represents “Web-Mining” class label. The results show that the source codes of these documents contain the total of (35,232,721) characters, (33,834,862) are noise characters and (1,397,859) are normal characters. Table 6 shows the distribution rate of these characters.

Table 6. The distribution rate of noise characters for (30) PDF document

| 30 PDF documents | Total characters | Percentage Rate |
|------------------|------------------|-----------------|
| Characters No.   | 35,232,721       | 100             |
| Characters as noise | 33,834,862     | 96.03           |
| Characters after noise cleaning | 1,397,859     | 3.97            |

To extract text from HTML Web documents, a stream of characters is converted into a stream of terms to represent a document. This issue requires eliminating the superfluous characters in HTML source code. An experiment is done by using a collection of (60) HTML Web documents for the “Web-Mining”, “Machine Learning”, “Neural Networks” and “Image-Processing”. The results show that the source code of these documents contains the total of (3,137,261) characters, (2,801,120) characters represent superfluous characters and (336,141) are normal characters, as shown in table (7)

Table 7. Distribution of characters for (60) HTML documents

| 60 HTML documents | Total characters | Percentage Rate |
|-------------------|------------------|-----------------|
| Characters No.    | 3,137,261        | 100             |
| Characters as noise | 2,801,120     | 89              |
| Characters after noise cleaning | 336,141     | 11              |

6.2.2. Frequent wordsets extracting results

After cleaning the text we utilizing Frequent Wordsets Mining Algorithm (FIMA) to generate frequent wordsets as shown in table 8 we obtained 8-wordsets for “Web-Mining” class label.

Table 8. Wordset for Web-Mining” class

| Wordsets | No. of generated sets |
|----------|-----------------------|
| 1-wordsets | 112                   |
| 2-wordsets | 499                   |
| 3-wordsets | 715                   |
| 4-wordsets | 513                   |
| 5-wordsets | 225                   |
| 6-wordsets | 45                    |
| 7-wordsets | 8                     |
| 8-wordsets | 1                     |
6.2.3. PDF and HTML classification results

After applying the proposed method to a number of PDF and HTML test documents related to “WebMining”, “Machine-Learning”, “Neural-Networks” and “Image-Processing” class labels, it gives high-quality results. Table 9 illustrates the classification results for all class labels.

| Class label          | Classification results | Relevant | Non-relevant | Total |
|----------------------|------------------------|----------|--------------|-------|
| Web-Mining           | Classified             | 41       | 2            | 43    |
|                      | Unclassified           | 1        | 116          | 117   |
| Machine Learning     | Classified             | 32       | 1            | 33    |
|                      | Unclassified           | 7        | 120          | 127   |
| Neural Networks      | Classified             | 35       | 0            | 35    |
|                      | Unclassified           | 2        | 122          | 124   |
| Image Processing     | Classified             | 40       | 4            | 44    |
|                      | Unclassified           | 0        | 112          | 112   |

By using the same experiment classification results for four class labels, precision, recall, and accuracy can be calculated. Table 10 illustrates these results.

| Class label         | Precision%  | Recall%   | Accuracy% |
|---------------------|-------------|-----------|-----------|
| Web-Mining          | 0.9534883   | 0.9761904 | 0.9647058 |
| Machine Learning    | 0.9696969   | 0.8205128 | 0.8888888 |
| Neural Networks     | 1           | 0.9459459 | 0.9722221 |
| Image Processing    | 0.9090909   | 1         | 0.9523809 |
| Average             | 0.9580690   | 0.9356622 | 0.9467330 |

7. Conclusions

In this paper, we proposed a general association rules-based machine learning algorithm, which dedicated to text classification. The proposed algorithm avoids us redesigning the learning algorithm with each type of text documents that leads extra costs of effort and time in addition to that the redesigned algorithm may not guarantee the accurate results and this is the most important reason for designing the proposed algorithm. Various types of documents were used to assess the proposed algorithm without any adaptation. The proposed algorithm showed a high classification accuracy. Document type-independence property of the algorithm reduces the effort to construct the classifier but may increase the work to extract the text from the documents according to their types. Therefore, preprocessing plays an important role in preparing the text for learning or classification. The large value of minimum support yields a small number of rules and vice versa. It is important using balanced minimum support and minimum confidence weights for generating desirable results.

References
[1] Du, K.-L., Swamy and M. N. S. 2014 Neural Networks and Statistical Learning.
[2] Al-radaideh Q., Al-shawakfa E., Ghareb, A. S. and Abu-salem H. 2011 An Approach for Arabic Text Categorization Using Association Rule
[3] Joachims, T. 1998 Text categorization with support vector machines: learning with many relevant features. Proceedings of the 10th European Conference on Machine Learning, pp. 137–142
[4] Jingnian Chen, Houkuan Huang, Shengfeng Tian and Youli Qu 2009 Feature selection for text classification with Naïve Bayes Expert Systems with Applications 36, 5432–5435.
[5] Vishwanath Bijalwan, Vinay Kumar, Pinki Kumari and Jordan Pascual 2014 KNN based Machine Learning Approach for Text and Document Mining International Journal of Database Theory and Application Vol.7, No.1, pp.61-70.
[6] Arlina D’cunha and Dr.A.K.Sen 2015 Hierarchical Approach for Scientific Document Classification International Conference on Computing, Communication and Automation.
[7] Kewen Chen, Zuping Zhang, Jun Long and Hao Zhang 2016 Turning from TF-IDF to TF-IGM for term weighting in text classification Expert Systems With Applications 66 pages 245–260
[8] Lucas Dixon, John Li and Jeffrey Sorensen 2018 Measuring and Mitigating Unintended Bias in Text Classification AIES 18, February 2–3, 2018, New Orleans, LA, USA
[9] Cynthia Van, Hee Els Lefever and Veronique Hoste 2018 ‘Ironic Detection in English Tweets Proceedings of the 12th International Workshop on Semantic Evaluation, pages 39–50
[10] Xilun Chen, Yu Sun and Ben Athiwaratkun 2018 Adversarial Deep Averaging Networks for CrossLingual Sentiment Classification Transactions of the Association for Computational Linguistics, 6, 557–570
[11] Hadi W., Al-Radaideh, Q. A., and Alhawari S. 2018 Integrating associative rule-based classification with Naïve Bayes for text classification. Applied Soft Computing, 69, 344– 356
[12] Zheng W. and Jin M. (2019). Comparing multiple categories of feature selection methods for text classification, Digital Scholarship in the Humanities.
[13] Hotho A, Nürnberger A and Paaß G 2005 A brief survey of text mining Ldv Forum 20(1)19-62.
[14] Al-Harbi S, Almuhareb A, Al-Thubaity A, Khorsheed MS and Al-Rajeh A 2008. Automatic Arabic text classification. JADT 77-83.
[15] R Agrawal, Tmielinski and A Swami 1993 Database Mining: A Performance Perspective IEEE Transactions on Knowledge and Data Engineering, 12:914-925.
[16] H.Wang, D. Can, A. Kazemzadeh, F. Bar, and S. Narayanan 2012 A System for Real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle Jeju, Repub. Korea, pp. 115–120.
[17] R.Agrawal and R. Srikant 1994 Fast Algorithms for Mining Association Rules in Large Databases J.Comput. Sci. Technol., vol. 15, no. 6, pp. 487–499
[18] V. Singh and B. Saini 2014 an Effective Tokenization Algorithm for Information Retrieval Systems pp. 109–119.
[19] Martin Porter 2006 Porter Stemming Algorithm, Available:https://tartarus.org/martin/PorterStemmer/index.html.
[20] Twitter Sentiment Analysis Training Corpus (Dataset), http://thinknook.com/twitter-sentiment-analysis-training-corpusdataset-2012-09-22