Topical News Classification Using Machine Learning Techniques

Edoh Audu Okechalu, Umoru Yahaya Ibrahim and Hashim Ibrahim Bisallah

August 21, 2024

TOPICAL NEWS CLASSIFICATION USING MACHINE LEARNING TECHNIQUES

Okechalu Edoh Audu
Ibrahim Umoru Yahaya
Hashim Bisallah Ibrahim

University of Abuja
Nigeria
okechalu@yahoo.com
**Okechalu Audu Edoh*

ABSTRACT
News is information that is presented through print, broadcast, Internet, or from mouth to mouth. For the ease of news, we classify news based on different category to help users to find relevant news rapidly. This classification results in the use of classifier engine to split any news into the respective category. This research employs the use of Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB) and Support Vector Machine (SVM) to classify topival news. The aim of this research is to develop a framework to categorize news topics in various categories and the objectives of this work are to pre-process the data using Term Frequency Inverse Document Frequency (tf-idf) and Bag of Words (BoW) which is suitable for the input to the classifier, apply Machine Learning (ML) techniques on pre-processed data, evaluate the performance of the machine learning classifiers on the pre-processed data and obtain the highest accuracy of the machine learning classifiers suitable on the pre-processed data. The finding shows SVM is a better classifier than NB, RF and DT using TFIDF while NB is a better classifier than SVM, RF and DT using BoW. Also, SVM is a better classifier using large datasets while NB thrives better with a smaller datasets.

Keywords: ML Classifiers, BoW, TF-IDF, NB,SVM.

1.0 INTRODUCTION

News, disseminated via various mediums such as print, broadcast, and online platforms, holds significant importance for individuals and communities alike, serving as a vital source of information (Dewi et al., 2011). However, the sheer volume of available articles online makes it challenging to efficiently locate relevant content. Thus, there is a growing need for automated news classification systems to categorize news articles swiftly and accurately. Data mining encompasses two primary techniques: unsupervised and supervised learning. News classification falls within the realm of supervised learning, where models are trained to categorize data. Various methods, including neural networks, decision trees, clustering, and naïve Bayes classifiers, are employed for news classification. Text mining, a burgeoning field, aims to extract meaningful insights from unstructured text. This involves analyzing large volumes of natural language text to identify patterns and extract valuable information (Mark et al., 2015). Document mining, a subset of text mining, focuses on extracting high-quality information from document collections such as news feeds and databases (Prabha & Suganya, 2017). The proliferation of digital news across diverse topics presents a challenge in categorizing news articles efficiently. Utilizing machine learning techniques, particularly classifier engines, facilitates the automatic categorization of news articles into relevant topics, thereby aiding users in accessing pertinent information quickly. Recent advancements in information technology have led to an exponential increase in available information, with news articles comprising a significant portion of factual data. Classifying news articles into appropriate categories remains a critical task to streamline information retrieval. Machine learning techniques are instrumental in achieving this objective by automating the classification process (Marty et al., 2017). Various studies have explored the application of Random Forest (RF) algorithms across different domains, showcasing its effectiveness in predictive modeling (Malek et al., 2018; Wang et al., 2018; Malazi & Davari, 2018; De Santana et al., 2018; Anitha & Siva, 2018). Researchers have optimized RF algorithms to enhance predictive capabilities for specific datasets, introducing variations such as Class Incremental RF (CIRF) and Random Trust RF (RTRF) to address specific challenges and improve performance in various scenarios (Hu et al., 2018; Abellán et al., 2018; Gomes et al., 2017; Genuer et al., 2017; Zhu et al., 2018).

2.0 *Related Concept*

2.1 *Multiple Instance Learning (MIL)*

Usually formulated as one of two SVM-based methods (mi-SVM and MI-SVM), multiple instance learning (MIL) is a supervised learning technique (Doran, G., & Ray, S. 2014). Rather than accepting instances as input, MIL accepts a set of labelled bags. The MIL method operates on the assumption that the input is provided as a collection of input patterns $(x1,\ldots,xn)$ organised into bags $(B1,\ldots,Bm)$ with $BI = \{xi : i \in I\}$ for the specified index sets $I \subsetneq \{1,\ldots,n\}$. Label YI is linked to every bag in BI; if xi in BI has at least one occurrence of a positive label, then $YI = 1$ and if $yi = -1$ for every i in I.

YI = max i∃yi or a series of linear constraints can be used to represent the relationship between instance labels yi and bag labels YI: ∑ yi +1 2 ≥1, i∃∀ s.t.YI = 1, yi = −1,∀ s.t.YI = −1.

2.2 ***Stacking Support Vector Machine (SSVM)***

A hierarchical classification technique called stacking support vector machine (SSVM) is applied to category tree structure using a top-down, level-based methodology (Singh et al., 2018). In general, this method yields more accurate results than single-SVM models since it offers a hierarchical model of individual SVM classifiers (Kowsari et al., 2019). A hierarchical classifier with many layers—two levels, or the main domain and sub-domains—is used by the stacking model.

2.3 *String Kernel*

The string kernel has also been used to study text classification. Using Φ(.) to map the string in the feature space is the fundamental concept behind the string kernel (SK). Text, DNA, and protein classification are just a few of the several applications that have used the spectrum kernel as part of SK (Singh et al., 2018). Counting the occurrences of a word in string xi as a feature map where creating feature mappings from x →Rlk is the fundamental principle behind the spectrum kernel. When it comes to classifying string sequences, SVM's primary drawback is its temporal complexity (Singh et al., 2018).

2.4 *Class SVM*

Let x1, x2,..., xn be training instances for class X in the context of text classification; X is a compact subset of RN. For multi-class situations, we must create a Multiple-SVM (MSVM), as SVMs are often employed for binary classification (Rosales-Pérez et al., 2018).

3.0 METHODOLOGY

An extensively used benchmark for text classification tasks is the 20 Newsgroups dataset. It is composed of about twenty newsgroups with about twenty thousand news articles each. Gathering and preprocessing data. Early in the 1990s, Usenet newsgroups were the source of the 20 Newsgroups dataset.In order to get the data ready for machine learning models, it goes through a number of preprocessing steps. These include the elimination of stop words, the removal of common words that aren't useful for classification, tokenization, stemming/lemmatization, feature extraction, and the representation of documents as vectors using methods like TF-IDF, Feature Representation, Bag-of-words, Model Training and Evaluation. The preprocessed data can be used to train a variety of machine learning models, including: The Naive Bayes model is an effective and basic approach for text classification. High accuracy can be attained with support vector machines (SVM), however they may need more intricate parameter adjustment than other models for model evaluation, including accuracy, precision, recall, and F1 score. Modules of Architecture: Reads and preprocesses the data using a data loader. Feature Extractor: This tool pulls characteristics out of text data. Model: The text classification is carried out via a machine learning model. Loss Function: Evaluates the performance of the model and directs the optimisation procedure, Optimizer: Modifies the parameters of the model to enhance its efficiency. Examines the model's performance with data that hasn't been seen before.Figure 1 shows all the steps of the technique and the implemented steps are explained in the sub-sections:

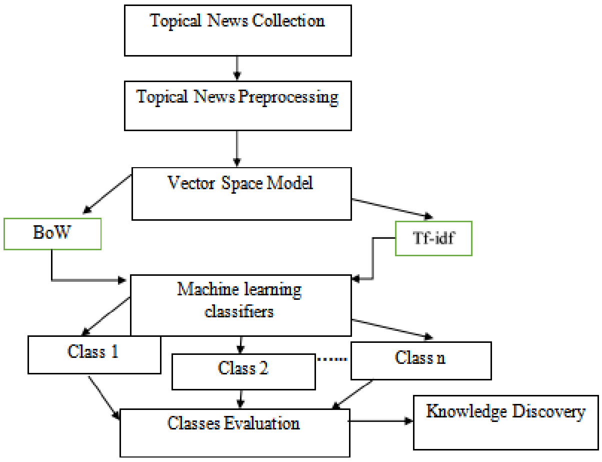

Figure 1: Methodology

4.0 EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Choice of Programming Language

The setup of the experiment in this research was implemented using python programming environment Anaconda 3.

4.2 Description of Dataset

One widely used benchmark for text classification tasks is the 20 Newsgroups dataset. It is composed of almost twenty thousand news items divided into twenty categories. A vector of word frequencies is used to represent each document, with each element denoting the frequency of a certain word inside the document. A more complex representation known as term frequency-inverse document frequency (TF-IDF) weights word frequencies according to their inverse document frequency. This lessens the weight given to terms that are common throughout the corpus but don't provide much information for specific texts.

The 20 Newsgroups dataset's features don't often have a strong correlation with one another. This is as a result of the features capturing various textual data characteristics. Word frequency, for instance, conveys the significance of a single word, but TF-IDF and n-grams show the connections between words. If stop words are left in, there is a positive association between word frequency and TF-IDF. However, the precise dataset and pre-processing techniques can affect the actual correlations.

This experiment was conducted on a known Real-world dataset because it is suitable for text classification technique. The dataset is obtained from UCI Machine Learning repository (https://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups). The detailed description of the dataset is as shown in Table 1.

Table 1: Description of dataset

Dataset	No. of samples	No. of dimensions	No. of class
20Newsgroups	19997	1000	20

4.3 Evaluation of Experiment

The evaluation of classification technique is based on the evaluation of the several classes involved in order to get accuracy. The high accuracy of any class is considered to be the best classification, hence a better performance will be achieved compared to the existing ones.

4.4 Discussions of the Results

The 20Newsgroup is implemented in Figure 2 and the results are shown in Figures 2-11:

```
In [ ]: DATA_DIR = '/Desktop/coding/20 Newsgroup Dataset'

In [ ]: texts[1]
```

Figure 2: Uploading and preprocessing the dataset

Figure 3 shows the first folder having 1000 features and the labels index having 20 folders. This represents that the 20Newsgroup dataset have 20 folders in it and each folder has about 1000 items.

```
In [ ]: labels[1000]
Out[56]: 1
```

```
In [ ]: labels_index
Out[55]: {'alt.atheism': 0,
 'comp.graphics': 1,
 'comp.os.ms-windows.misc': 2,
 'comp.sys.ibm.pc.hardware': 3,
 'comp.sys.mac.hardware': 4,
 'comp.windows.x': 5,
 'misc.forsale': 6,
 'rec.autos': 7,
 'rec.motorcycles': 8,
 'rec.sport.baseball': 9,
 'rec.sport.hockey': 10,
 'sci.crypt': 11,
 'sci.electronics': 12,
 'sci.med': 13,
 'sci.space': 14,
 'soc.religion.christian': 15,
 'talk.politics.guns': 16,
 'talk.politics.mideast': 17,
 'talk.politics.misc': 18,
 'talk.religion.misc': 19}
```

Figure 3: The dimensions (features) and topic categories (labels_index)

Figure 4 shows SVM classification when used on tf-idf.

```
[ ]: #svm evaluation on test data tfidf
pred_tf_test = model_tf.predict(X_test_tf)
print(classification_report(y_test_tf, pred_tf_test))
print()
print('Confusion Matrix: \n', confusion_matrix(y_test_tf, pred_tf_test))
print()
print('Accuracy :', accuracy_score(y_test_tf, pred_tf_test))
```

	precision	recall	f1-score	support
0	0.74	0.76	0.75	246
1	0.72	0.76	0.74	266
2	0.76	0.77	0.76	237
3	0.69	0.75	0.72	230
4	0.87	0.81	0.84	283
5	0.80	0.83	0.81	247
6	0.74	0.88	0.80	243
7	0.85	0.88	0.86	248
8	0.98	0.91	0.94	245
9	0.94	0.89	0.92	255
10	0.95	0.93	0.94	259
11	0.96	0.92	0.94	245
12	0.80	0.77	0.78	274
13	0.89	0.89	0.89	254
14	0.96	0.90	0.93	238
15	0.88	0.85	0.86	254
16	0.87	0.85	0.86	253
17	0.93	0.91	0.92	258
18	0.74	0.72	0.73	229
19	0.55	0.54	0.54	236
accuracy			0.83	5000
macro avg	0.83	0.83	0.83	5000
weighted avg	0.83	0.83	0.83	5000

Figure4: SVM evaluation report on tf-idf

Figure 5 shows NB classification when used on tf-idf.

```
[ ]: #naive bayes evaluation on test data tfidf
pred_naive_test = naive.predict(X_test_tf)
print(classification_report(y_test_tf, pred_naive_test))
print()
print('Confusion Matrix: \n', confusion_matrix(y_test_tf, pred_naive_test))
print()
print('Accuracy :', accuracy_score(y_test_tf, pred_naive_test))
```

	precision	recall	f1-score	support
0	0.71	0.78	0.75	246
1	0.78	0.70	0.74	266
2	0.71	0.74	0.72	237
3	0.60	0.80	0.69	230
4	0.91	0.75	0.82	283
5	0.74	0.84	0.79	247
6	0.84	0.74	0.78	243
7	0.88	0.88	0.88	248
8	0.93	0.91	0.92	245
9	0.94	0.91	0.92	255
10	0.93	0.97	0.95	259
11	0.89	0.93	0.91	245
12	0.84	0.72	0.77	274
13	0.95	0.87	0.91	254
14	0.95	0.90	0.92	238
15	0.72	0.94	0.82	254
16	0.84	0.91	0.87	253
17	0.88	0.95	0.91	258
18	0.69	0.73	0.71	229
19	0.59	0.33	0.43	236
accuracy			0.82	5000
macro avg	0.82	0.81	0.81	5000
weighted avg	0.82	0.82	0.81	5000

Figure 5: NB evaluation report on tf-idf

Figure 6 shows RF classification when used on tf-idf.

```
[ ]: pred = randclas.predict(X_test_tf)
print(classification_report(y_test_tf, pred))
print()
print('Confusion Matrix: \n', confusion_matrix(y_test_tf, pred))
print()
print('Accuracy :', accuracy_score(y_test_tf, pred))
```

	precision	recall	f1-score	support
0	0.63	0.57	0.60	261
1	0.63	0.57	0.60	277
2	0.60	0.76	0.67	230
3	0.62	0.60	0.61	247
4	0.81	0.78	0.79	254
5	0.67	0.71	0.69	246
6	0.59	0.73	0.65	260
7	0.75	0.78	0.77	253
8	0.90	0.86	0.88	232
9	0.84	0.85	0.84	238
10	0.87	0.93	0.90	261
11	0.90	0.86	0.88	247
12	0.68	0.57	0.62	262
13	0.79	0.85	0.82	245
14	0.83	0.85	0.84	239
15	0.65	0.83	0.73	242
16	0.73	0.81	0.77	246
17	0.88	0.86	0.87	251
18	0.65	0.54	0.59	234
19	0.37	0.22	0.28	275
accuracy			0.72	5000
macro avg	0.72	0.73	0.72	5000
weighted avg	0.72	0.72	0.72	5000

Figure 6: RF evaluation report on tf-idf

Figure 7 shows DT classification when used on tf-idf.

```
[ ]: pred_test = Decision_tree.predict(X_test_tf)
print(classification_report(y_test_tf, pred))
print()
print('Confusion Matrix: \n', confusion_matrix(y_test_tf, pred))
print()
print('Accuracy :', accuracy_score(y_test_tf, pred))
```

	precision	recall	f1-score	support
0	0.63	0.57	0.60	261
1	0.63	0.57	0.60	277
2	0.60	0.76	0.67	230
3	0.62	0.60	0.61	247
4	0.81	0.78	0.79	254
5	0.67	0.71	0.69	246
6	0.59	0.73	0.65	260
7	0.75	0.78	0.77	253
8	0.90	0.86	0.88	232

```
           9       0.84      0.85      0.84       238
          10       0.87      0.93      0.90       261
          11       0.90      0.86      0.88       247
          12       0.68      0.57      0.62       262
          13       0.79      0.85      0.82       245
          14       0.83      0.85      0.84       239
          15       0.65      0.83      0.73       242
          16       0.73      0.81      0.77       246
          17       0.88      0.86      0.87       251
          18       0.65      0.54      0.59       234
          19       0.37      0.22      0.28       275

    accuracy                           0.72      5000
   macro avg       0.72      0.73      0.72      5000
weighted avg       0.72      0.72      0.72      5000
```

Figure 7: DT evaluation report on tf-idf

Figure 8 shows SVM classification when used on BoW.

```
[ ]: #Testing the model with test set
     pred_test = model_k.predict(X_test)
     print(classification_report(y_test, pred_test))
     print()
     print('Confusion Matrix: \n', confusion_matrix(y_test, pred_test))
     print()
     print('Accuracy :', accuracy_score(y_test, pred_test))
              precision    recall  f1-score   support

           0       0.58      0.68      0.63       246
           1       0.60      0.62      0.61       266
           2       0.66      0.67      0.66       237
           3       0.52      0.60      0.56       230
           4       0.75      0.71      0.73       283
           5       0.80      0.72      0.75       247
           6       0.50      0.73      0.59       243
           7       0.74      0.75      0.74       248

           8       0.92      0.79      0.85       245
           9       0.87      0.75      0.81       255
          10       0.82      0.84      0.83       259
          11       0.84      0.80      0.82       245
          12       0.63      0.64      0.63       274
          13       0.83      0.68      0.75       254
          14       0.82      0.80      0.81       238
          15       0.78      0.72      0.75       254
          16       0.74      0.69      0.71       253
          17       0.85      0.81      0.83       258
          18       0.52      0.60      0.55       229
          19       0.40      0.37      0.38       236

    accuracy                           0.70      5000
   macro avg       0.71      0.70      0.70      5000
weighted avg       0.71      0.70      0.70      5000
```

Figure 8: SVM evaluation report on BoW

Figure 9 shows NB classification when used on BoW.

```
[ ]: pred_naive_test = naive_bow.predict(final_bow_test)
     print(classification_report(Ytest, pred_naive_test))
     print()
     print('Confusion Matrix: \n', confusion_matrix(Ytest, pred_naive_test))
     print()
     print('Accuracy :', accuracy_score(Ytest, pred_naive_test))
              precision    recall  f1-score   support

           0       0.66      0.67      0.66       257
           1       0.59      0.80      0.68       259
           2       0.79      0.65      0.71       243
           3       0.66      0.79      0.72       228
           4       0.92      0.64      0.75       250
           5       0.78      0.83      0.80       251
           6       0.92      0.51      0.66       271
           7       0.90      0.86      0.88       258

           8       0.97      0.90      0.93       240
           9       1.00      0.89      0.94       243
          10       0.95      0.97      0.96       249
          11       0.83      0.91      0.87       260
          12       0.87      0.64      0.74       242
          13       0.91      0.90      0.91       250
          14       0.90      0.86      0.88       258
          15       0.78      0.90      0.83       245
          16       0.80      0.87      0.83       252
          17       0.66      0.93      0.77       219
          18       0.57      0.78      0.66       276
          19       0.52      0.38      0.44       249

    accuracy                           0.78      5000
   macro avg       0.80      0.78      0.78      5000
weighted avg       0.80      0.78      0.78      5000
```

Figure 9: NB evaluation report on BoW

Figure 10 shows RF classification when used on BoW.

```
[11]: pred = randclas.predict(X_test)
      print(classification_report(y_test, pred))
      print()
      print('Confusion Matrix: \n', confusion_matrix(y_test, pred))
      print()
      print('Accuracy :', accuracy_score(y_test, pred))
              precision    recall  f1-score   support

           0       0.64      0.61      0.63       244
           1       0.61      0.70      0.65       236
           2       0.62      0.79      0.69       243
           3       0.71      0.71      0.71       241
           4       0.80      0.75      0.78       243
           5       0.79      0.77      0.78       248
           6       0.63      0.78      0.70       250
           7       0.81      0.80      0.80       250

           8       0.87      0.90      0.88       221
           9       0.81      0.89      0.84       247
          10       0.87      0.89      0.88       257
          11       0.91      0.89      0.90       261
          12       0.79      0.62      0.69       260
          13       0.86      0.79      0.82       277
          14       0.92      0.82      0.86       264
          15       0.68      0.88      0.77       234
          16       0.80      0.79      0.79       242
          17       0.87      0.89      0.88       257
          18       0.62      0.55      0.58       235
          19       0.46      0.30      0.36       290

    accuracy                           0.75      5000
   macro avg       0.75      0.76      0.75      5000
weighted avg       0.75      0.75      0.75      5000
```

Figure 10: RF evaluation report on BoW

Figure 11 shows DT classification when used on BoW.

```
[14]: pred_train = Decision_tree.predict(X_test)
      print(classification_report(y_test, pred_train))
      print()
      print('Confusion Matrix: \n', confusion_matrix(y_test, pred_train))
      print()
      print('Accuracy :', accuracy_score(y_test, pred_train))
              precision    recall  f1-score   support

           0       0.35      0.43      0.39       244
           1       0.40      0.40      0.40       236
           2       0.44      0.57      0.50       243
           3       0.43      0.45      0.44       241
           4       0.58      0.58      0.58       243
           5       0.51      0.53      0.52       248
           6       0.53      0.60      0.56       250
           7       0.59      0.55      0.57       250

           8       0.65      0.68      0.66       221
           9       0.69      0.65      0.67       247
          10       0.78      0.73      0.75       257
          11       0.73      0.67      0.70       261
          12       0.49      0.39      0.43       260
          13       0.61      0.57      0.59       277
          14       0.61      0.58      0.59       264
          15       0.48      0.55      0.51       234
          16       0.49      0.50      0.49       242
          17       0.71      0.67      0.69       257
          18       0.35      0.37      0.36       235
          19       0.20      0.15      0.17       290

    accuracy                           0.53      5000
   macro avg       0.53      0.53      0.53      5000
weighted avg       0.53      0.53      0.53      5000
```

Figure 11: DT evaluation report on BoW

There are four machine learning classifiers used on both tf-idf and BoW. Table 2 shows the comparison of the ML techniques used on tf-idf while Table 3 shows the comparison of the ML techniques used on BoW. Also, Figure 12 shows the percentage accuracy of each of the ML techniques used on tf-idf in bar chart while Figure 13 shows the percentage accuracy of each of the ML techniques used on BoW in bar chart.

Table 2: Comparison of machine learning techniques used on tf-idf

Comparison using tf-idf	**DT**	**NB**	**RF**	**SVM**
ACCURACY	72%	82%	72%	83%
PRECISION	72%	82%	73%	93%
RECALL	72%	81%	73%	91%
F1-SCORE	72%	82%	72%	92%

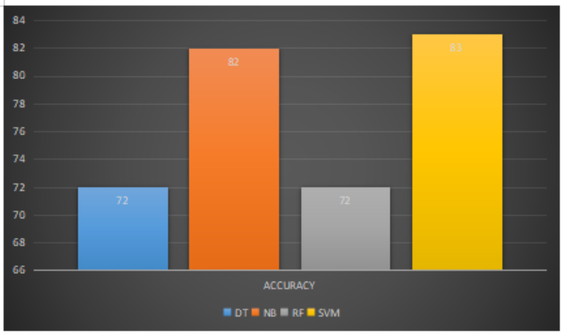

Figure 12: Machine learning techniques used on tf-idf

Table 3: Comparison of machine learning techniques used on BoW

Comparison using BoW	**DT**	**NB**	**RF**	**SVM**
ACCURACY	53%	78%	75%	70%
PRECISION	59%	80%	75%	71%
RECALL	59%	78%	76%	70%
FI-SCORE	59%	78%	75%	70%

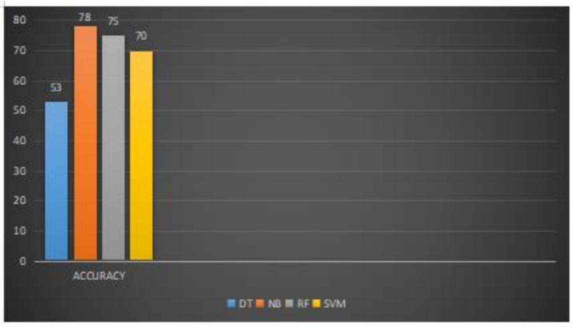

Figure 13: Machine learning techniques used on BoW

5.0 CONCLUSION

This research work reviewed several literatures related to news articles classification and separated into different news categories for ease of reading, researching and retrieving needed news. The experiment was setup using python programming language Anaconda 3.

Twenty newsgroups dataset was downloaded from UCI machine learning repository site as the preferred documents used in this research, the dataset was pre-processed before running through vector space model using BoW and TF-IDF. The four ML techniques (DT, NB, RF and SVM) acted upon the BoW and TF-IDF which classified the dataset into separate classes.

Evaluation of the performance of the four ML techniques used shows SVM has an accuracy of 83% using the vector space model of TF-IDF and it is suitable for use with large dataset. Also, NB has an accuracy of 82% using the vector space model of TF-IDF and it is suitable for use with small dataset.

REFERENCES

Abellán, J., Mantas, C.J., Castellano, J.G., & Moral-García, S. (2018). Increasing diversity in random forest learning algorithm via imprecise probabilities. Expert Systems with Applications, volume 97, pages 228–243.

Anitha, R. & Siva, S.R.D. (2018). Development of computer-aided approach for brain tumor detection using random forest classifier. International Journal of Imaging Systems and Technology, volume 28, no. 1, pages 48–53.

Aroor, D. D. & Sekhar, C. C. (2014). Class-specific GMM based intermediate matching kernel for classification of varying length patterns of long duration speech using support vector machines. Speech Communication, volume 57, pages 126–143.

De Santana, F.B., De Souza, A.M., & Poppi, R.J. (2018). Visible and near infrared spectroscopy coupled to random forest to quantify some soil quality parameters. Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy, volume 191, pages 454–462.

Dewi, Y. L., Agung, H., & Ridok, M. (2011). Indonesian News Classification Using Support Vector Machine. World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering, Volume 57, No. 9, Pages 767-770.

Doran, G., & Ray, S. (2014). A theoretical and empirical analysis of support vector machine methods for multiple-instance classification. Machine Learning. 97. https://doi.org/10.1007/s10994-013-5429-5.

Genuer, R., Poggi, J., Tuleau-Malot, C., & Villa-Vialaneix, N. (2017). Random forests for big data. Big Data Research, volume 9, pages 28–46.

Gomes, H.M., Bifet, A., & Read, J. (2017). Adaptive random forests for evolving data stream classification. Machine Learning, volume 106, no. 9-10, pages 1469–1495.

Hu, C., Chen, Y., Hu, L., & Peng, X. (2018). A novel random forests based class incremental learning method for activity recognition. Pattern Recognition, volume 78, pages 277–290.

Kowsari, K., Jafari M.K., Heidarysafa, M., Mendu, S., Barnes, L., Brown, D., Laura, I., & Barnes, D.

(2019). Text Classification Algorithms: A Survey. Information (Switzerland). https://doi.org/10. 10.3390/info10040150.

Malazi, H.T. & Davari, M. (2018). Combining emerging patterns with random forest for complex activity recognition in smart homes. Applied Intelligence, volume 48, no. 2, pages 315–330.

Malek, S., Gunalan, R., & Kedija, S. (2018). Random forest and Self Organizing Maps application for analysis of pediatric fracture healing time of the lower limb. Neurocomputing, volume 272, pages 55–62.

Mark, A. W., Ryan, K. L., & Ian, H. W. (2015). Bin Encoding: A User-Centric Secure Full-Text Searching Scheme for the Cloud. Trust Com/Big Data SE/ISPA (1): 563-570.

Marty, Emmanuel, Nathalie P. C., & Brigitte S. (2017). “Internet Users’ Participation and News Framing: The Strauss-Kahn Case–Related Live Blog at Le Monde.fr”. New Media and Society 19 (12): 1964–1982.

Prabha, K & Suganya. T. (2017). "A Guesstimate on Web Usage Mining Algorithms and Techniques," International Journals of Advanced Research in Computer Science and Software Engineering, volume 7, no. 6, pages 518-521.

Rosales-Pérez, A., Garcia, S., Terashima-Marín, H., Coello, C., & Herrera, F. (2018). Multiclass Classification Based on Cooperative Evolution of Support Vector Machines. IEEE Computational Intelligence Magazine. 13. 18-29. https://doi.org/10.1109/MCI.2018.2806997.

Singh, R., Sekhon, A., Kowsari, K., Lanchantin, J., Wang, B., & Qi, Y. (2018). A Fast GApped k-mer string Kernel using COunting. https://doi.org/10.1101/329425.

Wang, S., Liu, X., Yang, T., & Wu, X. (2018). Panoramic crack detection for steel beam based on structured random forests. IEEE Access, volume 6, pages 16432–16444.

Zhu, M., Xia, J., & Jin, X. (2018). Class weights random forest algorithm for processing class imbalanced medical data. IEEE Access, volume 6, pages. 4641–4652.