Performance evaluation of Machine learning algorithms in Biomedical Document Classification

Article · May 2020

CITATIONS
0

READS
123

2 authors:

Bichitrananda Behera
Pondicherry University
6 PUBLICATIONS **2** CITATIONS
SEE PROFILE

G. Kumaravelan
4 PUBLICATIONS **4** CITATIONS
SEE PROFILE

International Journal of Advanced Science and Technology
Vol. 29, No. 5, (2020), pp. 5704 - 5716

Performance evaluation of Machine learning algorithms in Biomedical Document Classification

Bichitrananda Behera[1], G.Kumaravelan[2]

Department of Computer Science, Pondicherry University, Puducherry, India
[1]bbehera19@gmail.com, [2]gkumaravelanpu@gmail.com

Abstract

Document classification is a prevalent task in Natural Language Processing (NLP) with a broad range of applications in the biomedical domain. In biomedical engineering categorization of biomedical literature into predefined categories becomes a cumbersome task. Hence, building an automatic document classifier using Machine Learning (ML) algorithms for the biomedical databases emerges as a significant task among the scientific community. In addition, empirical evaluation of these state-of-the-art classifiers for biomedical document categorization also becomes a thrust area of research. Hence, this paper examines the deployment of the various forefront ML algorithms in automatic classification of benchmark biomedical datasets like Bio Creative Corpus III, Farm-Ads, and TREC 2006 genetics Track. Finally, the performance measures of the ML classifiers have been evaluated through standard classification metrics like accuracy, precision, recall, and f1-measure.

Keywords: *Machine learning, Deep learning, Text Mining, document classification.*

1. Introduction

Biomedical engineering introduces different innovative engineering techniques and materials in medicine and healthcare for the development of biomedical tools and technology. In the era of internet-connected devices in every minute, a tremendous amount of biomedical data is generated with the rapid growth biomedical technology. In particular, biomedical research publishes a large number of science journals in electronic text form, and the automatic classification of these biomedical documents leads to a cumbersome task. Hence, building a classifier for large-scale biomedical documents plays a vital role. Nevertheless, text document classification is a prevalent task in NLP with broad applications in the biomedical domain [1], including biomedical literature indexing [2], automatic diagnosis codes assignment[3], tweets classification for public health topics [4], patient safety reports classification[5].

In general, an automatic document classification algorithm assigns a predefined label to the instances of the text documents (test data set) based on the classifier developed using ML/DL algorithm. However, the classifier captures the inherent patterns and relationships from the training data set. The most prominent ML classifiers [6] found in the literature are Decision Tree (DT), k-Nearest Neighborhood (KNN), Rocchio(RC), Ridge, Multinomial Naïve Bayes(M_NB), Bernoulli Naïve Bayes(B_NB) classifier, Support Vector Machine (SVM), Passive-Aggressive(PA) classifier, Random forest(RF), Artificial Neural Network (ANN) including Perceptron (PPN), Stochastic Gradient Descent(SGD) and Back Propagation neural network(BPN).

In the literature, only a few research attempts have been carried out to empirically evaluate the ML algorithms on the benchmark biomedical dataset in one platform. *A. M. Cohen* developed a replacement classification algorithm by assembling SVM with rejection sampling and chi-square feature selection technique for automatic

document classification [7]. The TREC 2005 genomics track biomedical dataset was used to compare the classification performance of the classifier with a different variant of SVM classifier. *H. Almeida, M.J. Meurs, L. Kosseim, G. Butler and A. Tsang* conferred supervised machine learning approaches like Naive Bayes, Support Vector Machine and provision Model Trees to perform text classification of PubMed abstracts, to support the triage of documents [8].

A bag-of-concepts representation of documents has been developed and applied machine learning algorithm like SVM for biomedical document classification [9]. *D.B. Nguyen, M. Shenify and H. Al-Mubaid* proposed an improved feature weighting technique for document representation and SVM as a classifier [10]. The proposed document representation technique provides the best classification performance compared to the documents represented in bag-of-words or TF-IDF document representation.

Therefore, the primary aim of this paper is to perform an end-to-end performance analysis of all the prominent ML algorithms deployed in automatic document classification of biomedical literature. Besides, the performance measures of the built-in classifiers are compared and empirically evaluated using well-defined metrics such as *accuracy*, *precision*, *recall*, and *f-measure* on the publicly available different biomedical datasets (BioCreative Corpus III(BC3), Farm-Ads, and TREC 2006 Genomics Track).

2. An Automated Biomedical Text Classification Process

In general, biomedical text documents constitute unstructured text documents from different biomedical repositories like *PubMed* and *MEDLINE*, web blogs, e-newspapers, medical reports, and social media. *Figure 1* shows the overall architecture of an automatic biomedical text document classification process.

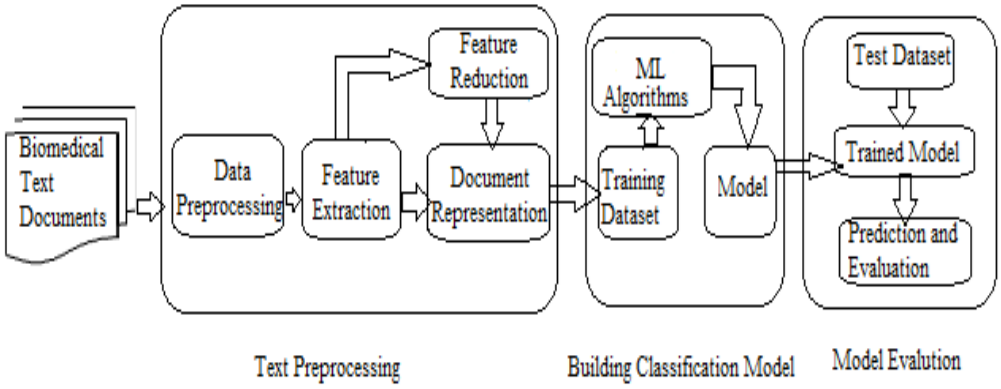

Figure 1. Biomedical Text Document Classification Process

Automatic biomedical text document classification consists of three major modules, namely, text preprocessing, building a classifier and model evaluation. The input to the text preprocessing module is the raw biomedical text documents from which it extracts valuable words or features and represents the documents in a suitable format through well-defined data preprocessing methods, feature extraction and feature reduction techniques. Methods such as *tokenization*, *stop word removal*, *lemmatization*, and *stemming* are used in data preprocessing to remove noisy and unwanted words from the documents to improve the performance of the classification model. *Bag-of-Words (BOW)*, *Term Frequency-Inverse Document Frequency (TF-IDF)*, *Word Embedding*, *Word2Vec*, *N-gram*, *Global Vectors* for Word Representation (*Glove*) and *FastText* are the feature

extraction methods used to extract the useful features from the documents for the matured document representation. Finally, feature reduction method includes feature selection/feature transformation techniques to minimize the text features without changing its indent meaning of the content.

The original idea behind the classification process is to construct a classification model, i.e., a classifier from the training dataset by relating the features within the text documents to one of the target class labels. Once the classifier is trained, it must own the predicting knowledge of assigning category labels to the test dataset. The model evaluation module includes the phenomenon of verifying the proof of correctness of the built classifier using the state-of-the-art measurements like *accuracy, error rate, precision, recall, and F-1 scores.*

3. ML Algorithms

3.1 Decision Tree

In the decision tree classification model, the instances are the documents and attributes of every document are itself a bag of words or terms. The decision tree classifier [11] performs hierarchical decomposition of training text documents by labelling its internal nodes with the text documents names, leaves of the tree with class labels and also the branches of the decision tree label with the test condition on terms. The test condition on terms could also be of two varieties depending on the document representation model. If the documents represent in the *Boolean document model*, then the first kind of test performed to determine the presence or absence of a selected term within the documents. In contrast, the second kind of test applicable to the terms if the document representation model is *TF-IDF* to looks at the weight of the terms within the text document.

The training phase uses different splitting criteria to build a decision tree from the training dataset. Most of the decision tree classifiers use a single attribute split combination wherever the one attribute is employed to perform the division [12]. If the information gain of an attribute or term is highest among all attributes then that attribute considers as a base node, and also the procedure is continual consequently for choosing the remaining nodes. Meanwhile within the testing phase, to predict the category label of a new untagged document $\boldsymbol{d_i}$, the decision tree classifier tests the terms of $\boldsymbol{d_i}$ against the decision tree ranging from the root node (base node) to a leaf node and assigns the category label of the leaf node to $\boldsymbol{d_i}$.

3.2 Naïve Bayes Classifier (NB)

Naïve Bayes classifier is a probabilistic classifier based on Bayesian posterior probability distribution. It holds the restriction with the independent relationship among the attributes through conditional probability. There is two variant of naïve Bayes classifier, namely the *multivariate Bernoulli model (B_NB)* and *multinomial model (M_NB)* [13]. The *multivariate Bernoulli naive Bayes model* works only on binary data. Hence, in document pre-processing steps, each attributes corresponding to the list of documents in *VSM* must be either one or zero depending on the presence or absence of that particular attribute in that document [14]. On the other hand, the *multinomial model* works on the frequencies of attributes available in *VSM* representation of the documents [15]. If the vocabulary size is small, then the *Bernoulli model* performs better than the *multinomial model.*

3.3 K-Nearest Neighbor Classifier (KNN)

Most of the classifiers within the literature pay longer in the training part for building the classification model are considered as an *eager learner*. However, k-NN classifier spends longer within the testing part for predicting the category label of the new untagged test document. Hence, it is known as a *lazy learner*. In the training section of the model

construction, k-nearest neighbor classifier stores all the training documents together with their target class. Meanwhile, in the testing phase, once any new test document comes for classification whose target class is unknown, k-nearest-neighborhood classifier finds the distance of the test document from all the training documents and assigns the category label of the training documents that is nearest or most like the unknown document [16]. For this reason, the k-nearest-neighborhood classifier is thought of as an *instant-based learning algorithm* [17]. Euclidian distance and cosine similarity are the foremost frequently used approaches for measurement similarity quotient to find the nearest neighborhood.

3.4 Support Vector Machine (SVM)

SVM is a kind of classifier has the potential to classify each linear and nonlinear data [18]. The core plan behind the SVM classifier is, it first non-linearly maps the initial training data into sufficiently higher dimension let be ***n***, so the data within the higher dimension is separated simply by ***n-1*** dimension decision surface known as *hyperplanes*. Out of all *hyperplanes*, the SVM classifier determines the simplest *hyperplane* that has most margins from the support vectors. Thanks to non-linearity mapping, SVM classifier works expeditiously on an oversized data set and has been with success applied in text classification [19].

3.5 Ridge Classifier (Ridge)

The Ridge classification algorithm relies on subspace assumption which states that samples of a specific class lie on a linear subspace and a new test sample to a category will be described as a linear combination of training samples of the relevant class [20]. The ridge classification algorithm is presented in *Figure 2*.

Input: Data matrix X holds training dataset and data matrix X-test holds the test dataset

Procedure:

Step-1: For each test data $x \in$ X-test, calculate the regression parameter vector $\hat{\alpha}$ as

$$\hat{\alpha} = \arg_{\alpha_i} \min \left\| x - X_i \alpha_i \right\|_2^2 + \lambda \left\| \alpha_i \right\|_2^2$$

where, λ the regularization parameter and i represent each class.

Step-2: Perform projection of new test sample x onto the subspace of each class i using $\hat{\alpha}_i$ as $\vec{x}_i = X_i \hat{\alpha}_i$

Step-3: Calculated distance between the test sample *x* and the class-specific sub-space $\vec{x}_i$.

Step-4: The test sample x is assigned to that class whose distance is minimum.

Figure 2. Ridge Classification Algorithm

3.6 Passive-Aggressive Classifier (PA)

The passive-aggressive classifiers belong to the family of large-scale learning algorithm [21]. The working principle of this kind of classifier is similar to that of Perceptron classifier; meanwhile, they do not require a learning rate. However, it includes a regularization parameter c. Figure 3. shows the pseudo-code description of the Passive aggressive classifier.

Algorithm: Passive-Aggressive (PA) classifier for multi-class classification

Input:

- D=<X, Y> is the dataset where X holds training instances and Y holds class labels
- Cost function $\rho(y, \bar{y})$

Initialize: Weight vector $w_1 = (0, ..., 0)$

1. **for** $i = 1, 2, ...$
2. Consider, instance $x_i \in X$ and its corresponding label $y_i \in Y$
3. **if** PA method == prediction-based(PB)
4. return $\bar{y}_i = \arg\max_{y \in Y}(w_i.\phi(x_i, y))$
5. **if** PA method==Max-loss(ML)
6. return $\bar{y} = \arg\max_{r \in Y}(w_i.\phi(x_i, r) - w_i.\phi(x_i, y_i) + \sqrt{\rho(y_i, r)})$
7. Loss function: $\ell_i = w_i.\phi(x_i, \bar{y}_i) - w_i.\phi(x_i, \bar{y}_i) + \sqrt{\rho(y_i, \bar{y}_i)}$
8. Compute: $\tau_i = \frac{\ell_i}{\|\phi(x_i, y_i) - \phi(x_i, \bar{y}_i)\|^2}$
9. Update weight $w_{i+1} = w_i + \tau_i(\phi(x_i, y_i) - \phi(x_i, \bar{y}_i))$

Figure 3. Pseudo Code for Passive-aggressive Classifier.

3.7 Artificial Neural Network (ANN)

ANN is a reasonably a data processing nonlinear model cherish the structure of the brain, and it will learn from the comprehensive training data to perform tasks like categorization, prediction or forecast, decision-making, visualization, and others. It consists of a compilation of nodes otherwise known as neurons that are the middle of data processing in ANN. With context to the problem statement, these neurons are organized into three different layers, specifically the input layer, an output layer, and hidden layer. Within the context of text classification, the quantity of words or terms outlines the neuron numbers within the input layer, and therefore the classes (class label) of documents define the number of neurons in the output layer. ANN will have a minimum of one input layer and one output layer; however, it is going to have several hidden layers relying upon the chosen drawback. All links from the input layer to the output layer through hidden layers are appointed with some weights that represent the dependence relation between the nodes. Once the neurons get weighted data, it calculates the weighted sum, and a well-known activation function processes it. The output value from the activation function is fed forward to all the neurons within the input layer to map the proper neuron in the output layer. Some examples of well-known activation functions are *Binary step*, *Sigmoid*, *TanH*, *Softmax*, and *Rectifier linear unit (ReLU)* functions. ANN can be additional versatile and more potent by employing additional hidden layers. In particular, *Perceptron (PPN)*, *Stochastic Gradient Descent (SGD)* neural network, and *Back-propagation neural network (BPN)* are the three widespread neural network primarily based classifiers that extensively used for text classification.

3.8 Rocchio Classifier (RC)

Rocchio classification algorithm is outlined on the conception of relevance feedback theory established within the field of Information Retrieval (IR) [22]. It uses the properties of centroid and similarity measure computations among the documents within the training and testing phase of model construction and usage, respectively. In the training phase, Rocchio classifier computes the centroid for each class from the relevant documents and establishes the centroid of each class as its representative. In the testing phase, to predict the category label of an untagged test document, Rocchio classifier calculates its Euclidean distance from the centroid of every class.

It assigns that class label which has a minimum distance from the untagged test document.

3.9 Random Forest

Figure 4 shows the overall working principle of random forest algorithm, which is a bagging type ensemble learning algorithm for the classification task. In the training phase, it builds several decision tree classifiers from the random sub-sample of documents. In the testing phase, each decision tree performs prediction for a new test document and assigns that class label, which is mostly predicted by all of the decision tree classifiers. The main advantage of random forest over the decision tree is that it eliminates the problem of over-fitting and increases the classification accuracy.

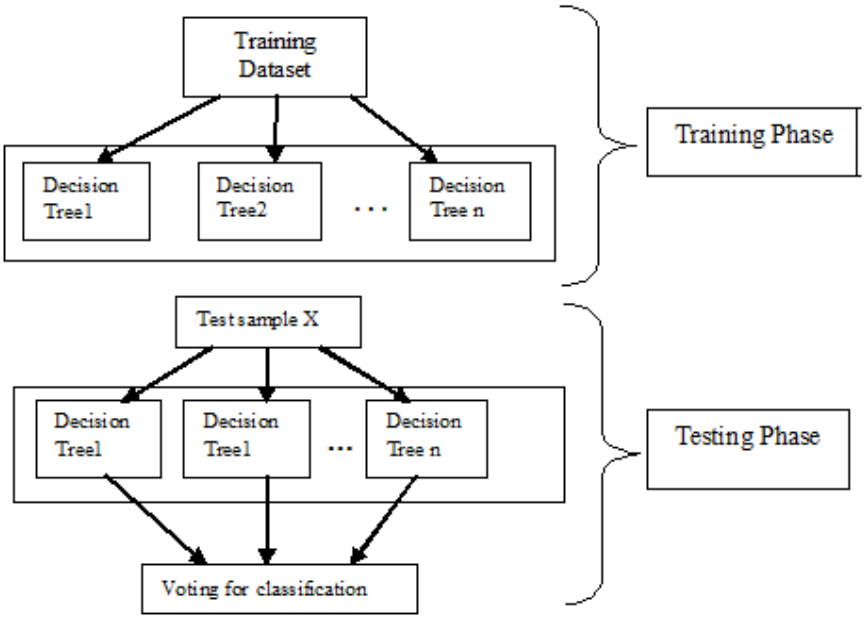

Figure 4. Random Forest classification

4. Result and Discussion

4.1. Experimental Setup

Scikit-learn Python libraries are used for the implementation of ML algorithms. End-to-end experimentation is performed using the cloud service, Google colaboratory having inbuilt GPU: 1xTesla K80, 2496 CUDA cores, RAM size 25GB and hard disk size 48.97GB. Table 1 depicts the summary of the four benchmark biomedical datasets, namely; two datasets of BioCreative Corpus III(BC3)[23] such as BC3-p1and BC3-p2, Farm-Ads dataset [24], and TREC 2006 Genomics Track[25].

- BioCreative Corpus III (BC3): The BC3 dataset has been created by the BioCreative III interactive task (IAT) of the BioCreative workshop that was conducted in 2010. The BC3 dataset is divided into BC3-part 1 and BC3-part 2 datasets. Both BC3-part 1 and BC3-part 2 datasets are initially in XML format and have size 32.5MB and 46.5MB, respectively. For document classification, all the abstract and respective class label of each document is extracted from the XML file and represent in a CSV file.
- Farm Ads dataset: This dataset contains 4142 number of farm ads text documents that represent various topics of farm animals. This is a binary classification problem where each of the documents or content either approves the ads or not. This dataset has a size 12.4MB and is available at the UCI machine learning repository.

- TREC 2006 Genomics Track dataset: This dataset is the collection of biomedical full-text HTML documents from 49 journals in the area of Genomics Track. In this experiment, a 1067 biomedical article abstract is collected from five journals, namely, Cerebral Cortex CC(201), Glycobiology GLY(203), Alcohol and Alcoholism AA(202), International Journal of Epidemiology IJE(206), and International Immunology II(265). A summary of the TREC dataset presented in Table 2.

Table1: Summary of four biomedical text dataset

Dataset	Classes	Number of Documents
BC3-p1	2	2280(1140+1140)
BC3-p2	2	3999(3317+682)
Farm-Ads	2	4142(1932+2210)
TREC	5	1067

Table 2.TREC 2006 Dataset Summary

Journal Name	No. of Documents
Cerebral Cortex CC	201
Glycobiology GLY	203
Alcohol and Alcoholism AA	202
International Journal of Epidemiology IJE	206
International Immunology II	206

4.2. Performance measure

In multiclass classification, performance measures such as Accuracy, Precision, Recall, and F1-Score are defined utilizing four features such as true Positive (tp_i), true Negative (tn_i), false Positive (fp_i), and false Negative (fn_i) of class C_i [26]. If m is the total number of classes in the dataset, then i value from 1 to m. There are three ways to calculate precision, recall, and F1-score over the whole test data: macro-averaged, micro-averaged, and weighted averaged. In this experiment, performance measures such as accuracy and weighted-average based precision, recall, and F1-score are used to evaluate the performance of the classifier. The performance measures are defined as follows.

$$Accuracy = \frac{\sum_{i=1}^{m} \frac{tp_i + tn_i}{tp_i + fp_i + fn_i + tn_i}}{m} \quad (1)$$

$$\Pr ecision_{Weighted} = \frac{\sum_{i=1}^{m} |y_i| \frac{tp_i}{tp_i + fp_i}}{\sum_{i}^{m} |y_i|} \quad (2)$$

$$\mathrm{Re}call_{Weighted} = \frac{\sum_{i=1}^{m} |y_i| \frac{tp_i}{tp_i + fn_i}}{\sum_{i}^{m} |y_i|} \quad (3)$$

$$F1-Score_{Weighted} = \frac{\sum_1^m |y_i| \frac{2tp_i}{2tp_i + fp_i + fn_i}}{\sum_i^m |y_i|} \quad (4)$$

4.3. Results

This section analyzes the result of the extensive experiment conducted on the machine learning (ML) algorithms on biomedical benchmark datasets like BC3-p1, BC3-p2, Farm-Ads and TREC. All the ML methods have implemented in the Scikit-learn ML library. The TfidfVectorizer of Python Scikit-learn envisages all the text

Table 3. Performance of ML algorithms for BC3-p1 dataset

Classification Algorithm	Performance Measure(Mean±Deviation)			
	Accuracy	Precision	Recall	F1-Score
KNN	0.5461±0.0514	0.5485±0.0527	0.5461±0.0514	0.5405±0.0521
DT	0.5276±0.0542	0.5271±0.0547	0.5276±0.0542	0.5242±0.0565
MNB	0.5894±0.0723	0.5957±0.0715	0.5895±0.0723	0.5769±0.0814
BNB	0.5996±0.1008	0.6042±0.1020	0.5995±0.1008	0.5841±0.1138
RF	0.6035±0.1101	0.6136±0.1133	0.6035±0.1101	0.5821±0.1275
SVM	0.6083±0.1132	0.6198±0.1155	0.6083±0.1132	0.5859±0.1304
PPN	0.5653±0.0451	0.5729±0.0516	0.5833±0.1051	0.5578±0.0499
SGD	0.5482±0.0610	0.5598±0.0650	0.5508±0.0894	0.5417±0.0639
Ridge	0.5754±0.0944	0.6059±0.1077	0.5868±0.1311	0.5605±0.1037
RC	0.6201±0.1290	0.6647±0.1454	0.6167±0.1916	0.5979±0.1461
PA	0.5557±0.0629	0.5749±0.0757	0.5500±0.0804	0.5480±0.0657
BPN	0.5675±0.0602	0.5857±0.0777	0.5710±0.0820	0.5600±0.0633

preprocessing routines to build a dictionary and finally to transform all the documents to *TF-IDF* [27] based *vector space model (VSM)* [28] document representation. In this experiment, the TF-IDF based VSM representation generates 11961, 17758, 26486 and 9676 number of features for BC3-p1, BC3-p2, Farm-Ads dataset and TREC dataset respectively. Once the features of the documents present

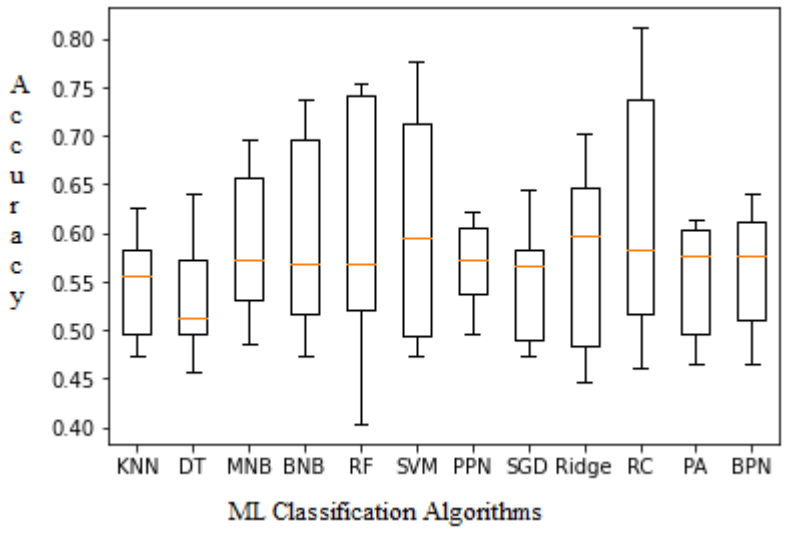

Figure 2. Box plot for Algorithm comparison on BC2-p1 dataset

in *TF-IDF* based *VSM document representation*, subsequently, training of all the ML algorithms performs with *10-fold cross-validation*. In *10-fold cross-validation*, the model training of ML algorithms performs in ten iterations. In every iteration, all the documents of the dataset equally divided into ten parts and each part select documents randomly from the whole dataset. Out of ten parts, nine parts usually consider for training, and one part uses for testing. After ten iterations, the mean and standard deviation of all the performance measures are evaluated. All the ML algorithms used the default hyper-parameters defined by *Scikit-learn* ML library. The classification performance with mean and deviation has shown below in *Table 3-6*. The classification accuracy of the ML algorithms has been compared and presented graphically in *Figure 2-5*.

Table 4. Performance of ML algorithms for BC3-p2 dataset

Classification Algorithm	Performance Measure(Mean±Deviation)			
	Accuracy	Precision	Recall	F1-Score
KNN	0.8334±0.0078	0.7948±0.0218	0.8334±0.0078	0.7860±0.0121
DT	0.8024±0.0144	0.8031±0.0148	0.8024±0.0144	0.8023±0.0135
MNB	0.8459±0.0099	0.8328±0.0119	0.8459±0.0099	0.8368±0.0107
BNB	0.8379±0.0165	0.8438±0.0171	0.8379±0.0165	0.8401±0.0157
RF	0.8347±0.0033	0.8478±0.0236	0.8347±0.0032	0.7656±0.0068
SVM	0.8465±0.0064	0.8558±0.0179	0.8464±0.0065	0.7922±0.0120
PPN	0.8462±0.0082	0.8360±0.0073	0.8462±0.0082	0.8390±0.0066
SGD	0.8562±0.0088	0.8394±0.0109	0.8562±0.0088	0.8394±0.0100
Ridge	0.8609±0.0088	0.8468±0.0142	0.8609±0.0088	0.8368±0.0116
RC	0.8026±0.0153	0.8555±0.0128	0.8026±0.0153	0.8193±0.0128
PA	0.8509±0.0086	0.8390±0.0074	0.8509±0.0086	0.8427±0.0075
BPN	0.8545±0.0098	0.8392±0.0116	0.8544±0.0098	0.8417±0.0109

From *Table 3*, it is clear that for BC3-p1 dataset the RC classifier performs best among all the classifiers with respect to all the classification performance measures. The classification accuracy of the RC classifiers is 0.6201±0.1290. The SVM classifier performs well next to RC classifier. After SVM, the RF and BNB classifiers perform well. If all the classifiers are compared with respect to the median of the boxplot has shown in *Figure 2*, then SVM and Ridge classifiers show best classification performance among all classifiers. However, Decision Tree classifier yields the lowest classification performance among all the classifiers for BC3-part 1 dataset. Meanwhile, the remaining classifiers provide an average classification performance.

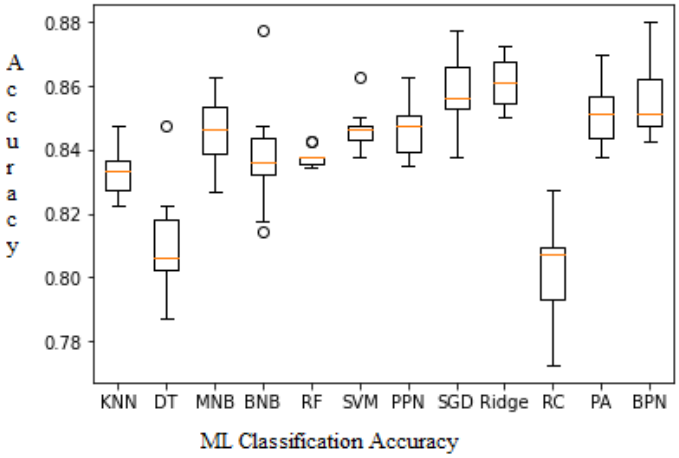

Figure 3. Box plot for Algorithm comparison on BC2-p2 dataset

Table 5. Performance of ML algorithms for Farm-Ads dataset

Classification Algorithm	Performance Measure			
	Accuracy	Precision	Recall	F1-Score
KNN	0.8788±0.0165	0.8796±0.0167	0.8788±0.0165	0.8785±0.0165
DT	0.8684±0.0076	0.8691±0.0077	0.8684±0.0076	0.8682±0.0077
MNB	0.9065±0.0090	0.9089±0.0086	0.9065±0.0090	0.9061±0.0091
BNB	0.8611±0.0189	0.8642±0.0168	0.8611±0.0189	0.8612±0.0189
RF	0.9043±0.0102	0.9074±0.0106	0.9043±0.0102	0.9038±0.0103
SVM	0.9123±0.0084	0.9149±0.0079	0.9123±0.0084	0.9119±0.0085
PPN	0.9000±0.0079	0.9010±0.0077	0.9000±0.0079	0.8998±0.0079
SGD	0.9048±0.0103	0.9052±0.0103	0.9048±0.0103	0.9047±0.0103
Ridge	0.9142±0.0073	0.9154±0.0075	0.9142±0.0073	0.9141±0.0073
RC	0.8686±0.0145	0.8735±0.0138	0.8686±0.0144	0.8675±0.0147
PA	0.9053±0.0061	0.9059±0.0061	0.9053±0.0061	0.9052±0.0061
BPN	0.9017±0.0091	0.9024±0.0092	0.9017±0.0091	0.9016±0.0091

In *Table 4*. For BC3-p2 dataset, the Ridge classifier followed by SGD classifier stands top among all the classifiers with respect to classification accuracy, precision, recall. On the other hand, SGD outperforms Ridge for F1 score. Next to Ridge and SGD, PA and BPN classifier performs well. Usually for BC3-part 2 dataset Decision Tree classifier generates lowest classification performance. In *Figure 3*, it is clear that concerning the median of classification accuracy, also Ridge, SGD, PA and BPN performs well.

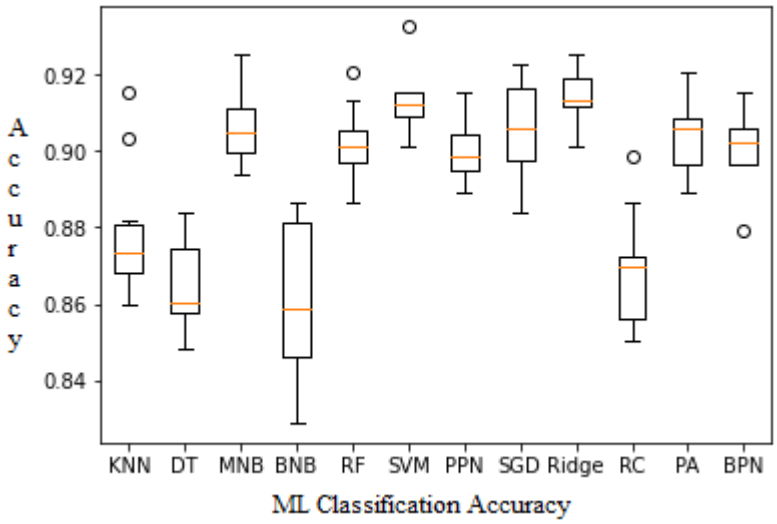

Figure 4. Box plot for Algorithm comparison on Farm-Ads dataset

From *Table 5* and *Figure 4*, it is clear that The Ridge classifier, followed by SVM, MNB and PA shows the best classification performance among all the classifiers for Farm ads dataset.

Table 6. Performance of ML algorithms for TREC dataset

Classification Algorithm	Performance Measure			
	Accuracy	Precision	Recall	F1-Score
KNN	0.8950±0.0319	0.9097±0.0270	0.8950±0.0319	0.8951±0.0316
DT	0.8510±0.0267	0.8574±0.0258	0.8510±0.0267	0.8512±0.0267
MNB	0.9634±0.0197	0.9647±0.0193	0.9634±0.0197	0.9630±0.0204
BNB	0.9588±0.0177	0.9613±0.0157	0.9588±0.0177	0.9584±0.0184
RF	0.9465±0.0188	0.9508±0.0176	0.9465±0.0188	0.9461±0.0192
SVM	0.9700±0.0154	0.9713±0.0148	0.9700±0.0154	0.9698±0.0156
PPN	0.9559±0.0183	0.9588±0.0170	0.9559±0.0183	0.9558±0.0183
SGD	0.9634±0.0193	0.9647±0.0188	0.9634±0.0193	0.9634±0.0191
Ridge	0.9765±0.0152	0.9777±0.0146	0.9765±0.0152	0.9765±0.0152
RC	0.9503±0.0213	0.9526±0.0201	0.9503±0.0213	0.9502±0.0214
PA	0.9756±0.0139	0.9770±0.0134	0.9756±0.0139	0.9755±0.0140
BPN	0.9700±0.0116	0.9712±0.0111	0.9700±0.0116	0.9699±0.0117

In *Table 6*, for TREC 2006 Genomics Track dataset, Ridge and PA classifier shows highest classification performance among all classifiers. If the median of classification accuracy is taken into consideration from boxplot of *Figure 5*, then Ridge classifiers perform better than PA. SVM and BPN classifiers have excellent classification performance next to PA. For TREC 2006 Genomics Track dataset DT classifier shows the least performance among all the classifiers.

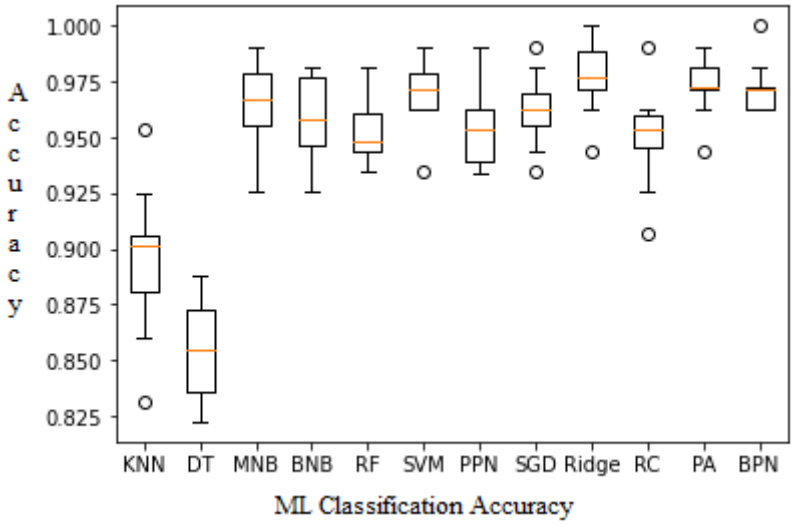

Figure 5. Box plot for Algorithm comparison on TREC dataset

Thus from the experimental analysis, it is clear that Ridge classifier followed by SVM performs almost best for all the benchmarking biomedical dataset. After Ridge and SVM, other classifiers like PA, SGD and BPN also provides excellent classification performance. The Decision tree and KNN provides least classification performance. The remaining classifiers provide average classification performance for biomedical datasets.

5. Conclusion and Future Scope

Medical document classification is a multidisciplinary field of research in biomedical engineering. Many supervised ML algorithms have been successfully applied for automatic classification biomedical literature. However, only a few authors addressed the performance measurements of all the classifiers in one platform. Hence, this research paper summarizes in detail the procedures involved automatic document classification process, exemplifies the working logic of the state-of-the-art supervised ML algorithms and empirically evaluates how all the ML algorithms which are constituted to act as a classifier to the benchmark biomedical dataset. Mainly, classifiers like Ridge, SVM, PA, SGD, and BPN provides excellent results on the given dataset compared to the other classifiers. However, the performance of KNN and Decision Tree classifiers has shown poor results for the chosen dataset compared to other classifiers. Meanwhile, other classifiers have an average classification performance. The future scope is to improve those classifiers to adapt well in connection to the large-scale dataset. As a result, application of deep learning-based models like multi-layer feed-forward neural networks, convolution Neural Networks (CNN), and recurrent Neural Networks (RNN) and ensemble deep learning models become an evitable avenue of further research.

References

[1] J. Du, Q. Chen, Y. Peng, Y. Xiang, C. Tao and Z. Lu, "ML-Net: multi-label classification of biomedical texts with deep neural networks." Journal of the American Medical Informatics Association 26, no. 11 (2019): 1279-1285.

[2] A. Neveol, S.E. Shooshan, S.M. Humphrey, J.G. Mork and A.R. Aronson, "A recent advance in the automatic indexing of the biomedical literature", Journal of biomedical informatics 42, no. 5 (2009): 814-823.

[3] S. V. Pakhomov, J. D. Buntrock, and C. G. Chute, "Automating the assignment of diagnosis codes to patient encounters using example-based and machine learning techniques" Journal of the American Medical Informatics Association 13, no. 5 (2006): 516-525.

[4] X. Ji, S. A. Chun, Z. Wei and J. Geller, "Twitter sentiment classification for measuring public health concerns", Social Network Analysis and Mining 5, no. 1 (2015): 13.

[5] Y. Wang, E. Coiera, W. Runciman and F. Magrabi, "Using multiclass classification to automate the identification of patient safety incident reports by type and severity", BMC medical informatics and decision making 17, no. 1 (2017): 84.

[6] B. Behera and G. Kumaravelan. "Towards the Deployment of Machine Learning Solutions for Document Classification", International Journal of Computer Sciences and Engineering, Vol.7(3), Mar 2019, E-ISSN: 2347-2693.

[7] A.M. Cohen. "An effective general purpose approach for automated biomedical document classification", In AMIA annual symposium proceedings, vol. 2006, p. 161. American Medical Informatics Association, 2006.

[8] H. Almeida, M.J. Meurs, L. Kosseim, G. Butler and A. Tsang, “Machine learning for biomedical literature triage”, Plos One. 2014, 9(12).

[9] M.A.M. García, R.P. Rodríguez and L.E.A. Rifón, “Biomedical literature classification using encyclopedic knowledge: a Wikipedia-based bag-of-concepts approach”, PeerJ. 2015.

[10] D.B. Nguyen, M. Shenify and H. Al-Mubaid, “Biomedical Text Classification with Improved Feature Weighting Method”, BICOB 2016, April 4-6 2016, Las Vegas, Nevada, USA.2016.

[11] Y. H. Li and A.H. Jain, " Classification of text documents”, The Computer Journal.1998, 41(8), 537–546.

[12] C.C. Aggarwal and C.X. Zhai, “Mining text data”, Springer. 2012.

[13] A. McCallum and K. Nigam, “A comparison of event models for naive bayes text classification”, In AAAI-98 workshop on learning for text categorization.1998, 752, 41–48.

[14] D.D. Lewis, “Naive (Bayes) at forty: The independence assumption in information retrieval”, In Machine learning: ECML-98, Springer. 1998, 4–15.

[15] A. McCallum, R. Rosenfeld, T.M. Mitchell and A.Y. Ng, "Improving Text Classification by Shrinkage in a Hierarchy of Classes", In ICML. 1998, 98,359–367.

[16] F. Sebastiani, "Machine Learning in Automated Text Categorization", ACM Computing Surveys. 2002, 34(1).

[17] E.S. Han, G. Karypis and V. Kumar, "Text categorization using weight adjusted k-nearest neighbor classification", Springer.2001.

[18] C. Cortes and V. Vapnik, " Support-vector networks", Machine Learning. 1995, 20, 273–297.

[19] H. Drucker, D. Wu, V. Vapnik, "Support Vector Machines for Spam Categorization", IEEE Transactions on Neural Networks. 1999, 10(5), 1048–1054.

[20] J. He, L. Ding, L. Jiang and L. Ma, "Kernel ridge regression classification", Proceedings of the International Joint Conference on Neural Networks.2014, 2263-2267.

[21] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz and Y. Singer, "Online passive aggressive algorithms", Journal of Machine Learning Research. 2006, 7,551–585.

[22] J.J. Rocchio, "Relevance Feedback in Information Retrieval" The SMART Retrieval System. 1971, 313–323.

[23] C.N. Arighi, P.M. Roberts and S. Agarwal et al., BioCreative III interactive task: an overview. BMC Bioinformatics 12, S4 (2011) doi:10.1186/1471-2105-12-S8-S4.

[24] M. Lichman, "UCI Machine Learning Repository Irvine", CA: University of California, School of Information and Computer Science, https://archive.ics.uci.edu/ml/datasets.html, 2013.

[25] W. Hersh, E. Voorhees. TREC genomics special issue overview. Journal Information Retrieval. Volume 12, Issue 1, Springer February 2009, p 1 - 15.

[26] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Inform. Process. Manage. 2009, 45(4), 427-437.

[27] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for IDF", Journal of documentation.2004

[28] G. Salton, A. Wong and C.S. Yang, "A vector space model for automatic indexing. Communications of the ACM 18, no. 11(1975): 613-620.