International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 64-69
Published Online June 2022 in IJEAST (http://www.ijeast.com)

MACHINE LEARNING PIPELINE FOR MULTI-CLASS TEXT CLASSIFICATION

Orobor, Anderson Ise,
Computer Science Department
Federal University of Petroleum Resources, Effurun, Delta State, Nigeria

Obi, Nneka Obiageli
Computer Engineering Department
Federal University of Petroleum Resources, Effurun, Delta State, Nigeria

***Abstract*— Machine Learning (ML) pipeline is a sequential step that orchestrates the flow of data from data pre-processing to model training and prediction. This paper presents the development of a ML pipeline based on Natural Language Processing (NLP) for multi-class text classification using the 20 newsgroups text dataset. The study experimented the performance of six classifiers which are Multinominal Naïve Bayes (MNB), Logistic Regression (LR), K Nearest Neighbors (KNN), Random Forest (RF), eXtreme Gradient Boosting (XGB), and Stochastic Gradient Descent (SGD) in Google Colab. Experimental results show that TF-IDF Vectorizer performed better than Count Vectorizer when used as the vectorizer in most cases. KNN consistently had the least performance in most of the cases. MNB and SGD had the best performance with an accuracy of 76% and 74% and a computation speed of 10min 14s and 1h 28min 21s respectively. The study suggests that improved accuracy can be obtained using a hybrid model or deep learning approach.**

***Keywords*— Natural Language Processing, 20 Newsgroups, Text Classification, Machine Learning, Pipeline**

I. INTRODUCTION

Research has demonstrated that information concealed in unstructured data can play an important part in decision-making. This significant portion of organisational information that is unstructured may contain the knowledge that is required for strategic planning [1]. The majority of the unstructured data are in form of text from different sources. Since analysing, interpreting, organising, and sorting through text data is difficult and time-consuming, several organisations do not use it to its full potential despite all of the inherent benefits. To begin to derive insight, a text classification approach can be adopted. The approach can be utilised to organise, arrange, and classify virtually any form of text, including documents, medical research, files, and text found all over the internet. The process of text classification is a method of Machine Learning (ML) that involves the assignment of a set of predetermined categories to free-form text. Text classification is one of the fundamental problems involved in Natural Language Processing (NLP), and it has a wide range of applications, including intent detection, topic labelling, spam detection, and sentiment analysis.

Text classification is more significant for many businesses since it reduces the need for manual data classification, which is a technique that is more expensive and takes more time to complete [2]. Text classification allows businesses to swiftly and cost-effectively organise all kinds of relevant content, such as emails, legal papers, social media, complaints, surveys, and many more sorts of content. It is one of the most important aspects of ML since it enables businesses to get profound insights that can guide future decision-making. The classification of text can either be done manually or automatically. Automatic text classification can be accomplished in three major ways, which are through machine learning-based systems, rule-based systems, or hybrid systems. In the process of manually classifying texts, a human annotator is required. This individual reads the text, analyses its meaning, and then assigns it to one of several categories. Although it is time-consuming and costly, this approach has the potential to produce satisfactory outcomes.

In this paper, we developed a ML pipeline for multi-class text classification. We experimented different ML classifier performances, explore common NLP techniques, and suggested potential approaches to developing a more accurate classifier.

The rest of the paper is organised as follows. Related works are discussed in section II. The research methodology is presented in section III. The experimental setup and results are presented in section IV. Section V is the discussion of the results obtained and the Conclusion is in section VI.

II. RELATED WORKS

The Support Vector Machines (SVM) was utilized in the process of classifying English text and documents, as described in [3] article. The author carried out two separate analytical tests with English documents in order to validate the classifiers that were chosen. Rocchio classifier produces the

best performance when the size of the feature set was small, but the SVM surpasses all of the other classifiers. The experiments were carried out on a set of 1033 text documents. Experimental results show that the classification rate increases to more than 90 percent when more than 4000 features are utilised.

[2] conducted a comparative study of text classification in which they studied and compared the effectiveness of several ML algorithms on various datasets. In the study, the authors focused on text classification. The work makes use of Support Vector Machine (SVM), k-Nearest Neighbor (k-NN), Logistic Regression (LR), Multinomial Naive Bayes (MNB), and Random Forest (RF). A comparison of these algorithms was carried out using two distinct datasets in order to get the most accurate results. According to the findings obtained from the proposed system, the results show that LR and SVM perform better than the other classifiers for the IMDB dataset, while kNN performs better than the other classifiers for the SPAM dataset.

In the work of [4], named entities were used as features for the purpose of classifying news articles into a pre-constructed hierarchy of international relations. The implementation of the feature selection was based on the named entities that were connected with the local categories. The authors trained SVM and according to the findings of the experiments, the use of named entities resulted in an improvement in the effectiveness of hierarchical text classification when applied to newspaper articles.

In their research, [5] investigated the problem of multi-class text classification for the Uzbek-language texts that were available. A dataset was developed specifically for the articles that were chosen from the online news version of the Uzbek newspaper "Daryo" consisting of 10 categories. SVM, Decision Tree (DT), RF, LR, and MNB were the six different machine learning techniques that were experimented with. As feature extraction methods, the TF-IDF algorithm, word-level, and character-level n-gram models, and character-level n-gram models were utilized and experimental results achieved maximum accuracy of 86.88 %.

[12] study introduced a research paper categorization system that can cluster research papers into the meaningful class in which papers are very likely to have similar subjects. The system can do this by using a clustering algorithm. By utilizing a Latent Dirichlet Allocation (LDA) scheme, the aforementioned system is able to extract representative keywords from the abstracts of each publication and topic. After that, the K-means clustering algorithm is used to categorize the full papers into research papers that have subjects that are comparable to one another. This classification is done based on the Term frequency-inverse document frequency (TF-IDF) values of each paper.

[13] study objective was to get over the limitations of single-label classification (SLC) and multi-label classification (MLC). The authors suggested an approach that leverages the Word2Vec paradigm for textual representation.

[14] suggest an innovative method for representing text texts that is based on an approach known as feature clustering. Textual records can be given a symbolic representation in the form of an interval-valued representation by utilizing the method that has been suggested. The authors evaluated the accuracy of classification gained by comparing it to the accuracy achieved by various current classifiers such as Naive Bayes, k-NN, Centroid based, and SVM classifiers. The findings of the experiments indicate that the acquired classification accuracy is superior to that of the approaches that are already in use.

The purpose of the study conducted by [15] was to apply ML techniques in order to automatically identify high-quality, content-specific articles for a given time period in the field of internal medicine and compare their performance with that of previous Boolean-based PubMed clinical query filters. The selection criteria used by the ACP Journal Club for publications in internal medicine served as the foundation for determining high-quality articles in the fields of etiology, prognosis, diagnosis, and therapy. According to the findings of the study, it is possible to automatically build models for the purpose of retrieving high-quality articles.

III. RESEARCH METHODOLOGY

The goal of the paper is to explore a basic workflow to train and evaluate a model capable of classifying text based on its content. The dataset used is "The 20 Newsgroups Text Dataset" which comprises over 18000 newsgroups articles on 20 topics that are originally spitted into training and testing sets [6]. These articles are unstructured in nature. In this paper, we will use a set of NLP techniques to prepare the dataset, training different supervised ML classifiers to classify articles and evaluate the models to determine the best performing classifier.

A. Data Visualisation

This gives a pictorial representation of the datasets available. This enables us to visualise concepts and data patterns. It gives us a clear idea of what the information means and also the distribution of data whether it is skewed towards any of the classes. Figure 1 shows the class distribution of the original dataset.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 64-69
Published Online June 2022 in IJEAST (http://www.ijeast.com)

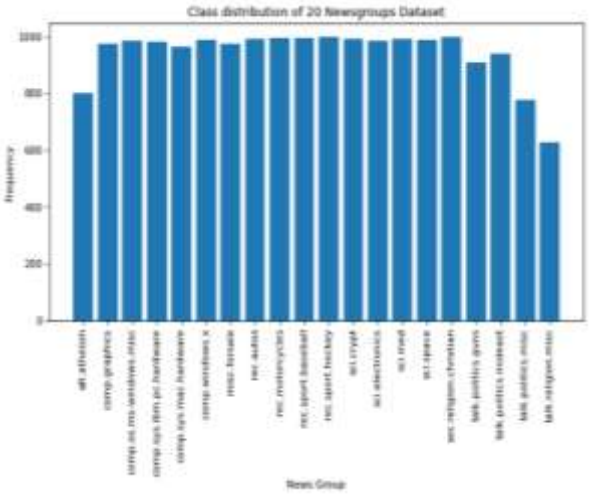

Fig 1. Dataset Distribution

From Fig 1, the output variable shows a near equal distribution hence, the dataset does not require resampling. Based on the even distribution, we are sure that our model will not be bias.
Fig 2 shows the total number of articles and the different classes/categories the dataset is organised. A sample article contained in the dataset is depicted in Fig 3.

```
Number of articles: 18846
Number of diffrent categories: 20
['alt.atheism',
 'comp.graphics',
 'comp.os.ms-windows.misc',
 'comp.sys.ibm.pc.hardware',
 'comp.sys.mac.hardware',
 'comp.windows.x',
 'misc.forsale',
 'rec.autos',
 'rec.motorcycles',
 'rec.sport.baseball',
 'rec.sport.hockey',
 'sci.crypt',
 'sci.electronics',
 'sci.med',
 'sci.space',
 'soc.religion.christian',
 'talk.politics.guns',
 'talk.politics.mideast',
 'talk.politics.misc',
 'talk.religion.misc']
```

Fig 2. Dataset Categories

```
From: gt@teal.csn.org (Eric H. Taylor)
Subject: Re: Gravity waves, was: Predicting gravity wave quantization & Cosmic Noise
Summary: Dong .... Dong .... Do I hear the death-knell of relativity?
Keywords: space, curvature, nothing, tesla
Nntp-Posting-Host: teal.csn.org
Organization: 4-L Laboratories
Distribution: World
Expires: Wed, 28 Apr 1993 06:00:00 GMT
Lines: 30

In article <C5nv2F.4op@well.sf.ca.us> metares@well.sf.ca.us (Tom van Flandern) writes:
>crb7@kelvin.seas.virginia.EDU (Cameron Randale Bass) writes:
>> Bruce.Scott@launchpad.unc.edu (Bruce Scott) writes:
>>> "Existence" is undefined unless it is synonymous with "observable" in
>>> physics.
>> [crb] Dong .... Dong .... Dong .... Do I hear the death-knell of
>> string theory?
>
>    I agree.  You can add "dark matter" and quarks and a lot of other
>unobservable, purely theoretical constructs in physics to that list,
>including the omni-present "black holes."
>
>    Will Bruce argue that their existence can be inferred from theory
>alone?  Then what about my original criticism, when I said "Curvature
>can only exist relative to something non-curved"?  Bruce replied:
>"'Existence' is undefined unless it is synonymous with 'observable' in
>physics.  We cannot observe more than the four dimensions we know about."
>At the moment I don't see a way to defend that statement and the
```

Fig 3. Sample Text Article

B. Data Pre-processing
Having visualised the dataset, we need to prepare the raw data and make it suitable for a ML model to accept. We performed two basic tasks here:

1. Data cleaning - this involves removing article metadata, converting text to lowercase, and removing stop words, alphanumeric characters, and punctuations. The article metadata was stripped out at the point of fetching the dataset by specifying a parameter remove. This enables us to avoid our classifiers from overfitting hence lacking the ability to generalised with other documents.
2. Data transformation – this entails transforming the cleaned text into numerical values that can be analysed statistically. This can be done using the sklearn API, which allows us to extract features that will be utilised to train the classifier. Two prominent approaches for feature extraction are TF-IDF Vectorizer and Count Vectorizer.

In addition to the above, we performed stemming, lemmatization, and ngram to determine their effect on the classifier performance.

C. Building a Pipeline
A ML pipeline is a sequential step that orchestrates the flow of data from data pre-processing to model training and prediction.
The automation of the ML model life cycle processes is the primary advantage that can be gained from utilising ML pipelines [11]. Scikit-learn provides a Class called Pipeline, which allows us to a create pipeline for a classifier. This has a very simplified interface that allows us to specify our vectorizer and classifier function as a parameter. We used the pipeline to chain part of the pre-processing task that is involved in training the classifier. We started our classification with one of the most common classifiers for discrete classification, MNB. By simply defining the classifier function in the pipeline as shown in Fig 4, five other classifiers were trained and evaluated.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 64-69
Published Online June 2022 in IJEAST (http://www.ijeast.com)

Fig 4. A Simple Pipeline

D. Model Evaluation

The model built in the previous section is evaluated at this stage to obtain the performances of the different classifiers. Standard metrics such as Accuracy, Receiver Operator Characteristic-Area Under Curve (ROC AUC) score, and Confusion Matrix are used for the evaluation. The various performances obtained from the experimentation processes are shown in Table 1 and Table 2. These results are based on our first attempt, in which the performances shown can be improved by performing hyperparameter tuning on the classifiers.

E. Grid Search for Parameter Tuning

All classifiers can be turned to obtain optimal performance through their parameters. At this stage of the evaluation, we created a list of parameters for each classifier. Scikit-learn GridSearchCV was used to fine-tune the parameters. The GridSearchCV searches and find the best hyperparameter value for the classifier. The performance of the various classifiers is shown in Table 3 after performing a grid search.

IV. EXPERIMENTAL SETUP AND RESULTS

The ML pipeline was developed in python programming language with extensive use of Sklearn libraries. The experimentation of the pipeline was performed on Google Colab. Colab is a platform developed by Google Research that makes it possible for anyone with access to the internet to write and run Python code directly within a web browser. The use of Colab is completely free, and it requires no installation or configuration on your local computer. Because Google hosts your Jupyter notebook and allows you to use their GPU at no additional cost, it is especially helpful for PCs that move at a snail's pace. It provides access to two key platforms for processing resources, namely the GPU and the CPU.

The experimental results are shown in Tables 1-3. Table 1 shows the different classifier performances on test data with and without parameter tuning. Fig 5 shows the confusion matrix of the best classifier.

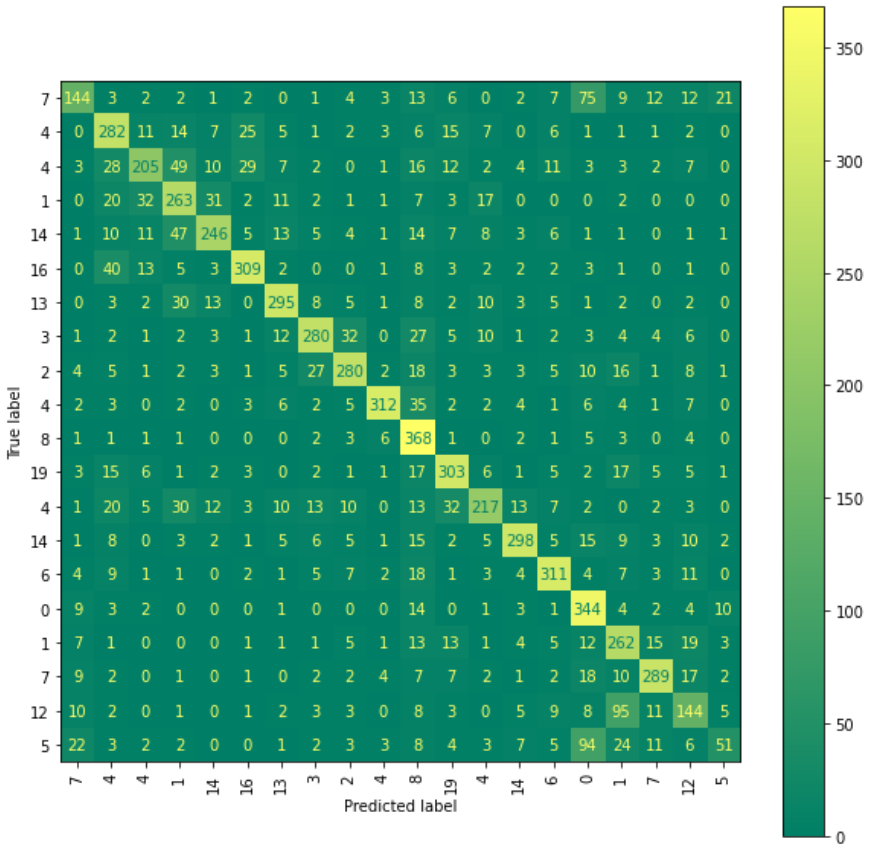

Fig 5. MNB Classifier Confusion Matrix

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 2, ISSN No. 2455-2143, Pages 64-69
Published Online June 2022 in IJEAST (http://www.ijeast.com)

Table 1. Classifiers Performances on Test Data

Classifiers						
Metrices	**MNB**	**RF**	**LR**	**KNN**	**XGB**	**SGD**
Using Count Vectorizer						
Accuracy (%)	58.28	58.43	60.07	16.31	58.20	55.27
ROC AUC score	0.90	0.91	0.92	0.61	0.92	-
Training Time	1.83 s	1min 1s	41 s	1.83 s	3min 45s	5.44 s
Using TF-IDF Vectorizer						
Accuracy (%)	60.51	58.92	67.18	08.44	57.68	69.19
ROC AUC score	0.94	0.91	0.94	0.53	0.92	-
Training Time	1.99 s	59.2 s	43.6 s	1.97 s	5min 30s	3.56 s
Using TF-IDF Vectorizer + Stopwords						
Accuracy (%)	67.51	62.08	68.64	07.59	58.15	69.25
ROC AUC score	0.95	0.93	0.95	0.52	0.92	-
Training Time	1.82 s	58 s	39.6 s	1.78 s	4min 12s	3.06 s
Using TF-IDF Vectorizer + Stopwords + Stemming						
Accuracy (%)	66.52	61.49	68.50	8.80	58.94	69.59
ROC AUC score	0.95	0.92	0.95	0.53	0.92	-
Training Time	48.5 s	1min 31s	1min 16s	48.8 s	4min 37s	49.4 s
Using TF-IDF Vectorizer + Stopwords + Lemmatization						
Accuracy (%)	66.87	62.02	68.37	6.87	59.14	69.64
ROC AUC score	0.95	0.93	0.95	0.52	0.92	-
Training Time	19.7 s	1min 8s	56.9 s	22.4 s	4min 25s	23.2 s

Table 2. Classifiers Performances on Test Data with Ngrams

	Classifiers Accuracy (%)					
ngrams	**MNB**	**RF**	**LR**	**KNN**	**XGB**	**SGD**
Unigram	65.34	61.88	68.57	06.54	58.68	69.41
Unigram+Bigram	65.57	62.50	68.26	06.49	58.40	70.43
Bigram	51.03	39.93	50.53	06.50	34.26	53.08

Table 3. Classifiers Performances on Test Data with Hyperparameter Tuning

	Classifiers					
Metrics	**MNB**	**RF**	**LR**	**KNN**	**XGB**	**SGD**
Best Score (%)	76	6	73	11	64	74
Best Params	alpha: 0.005	max_depth: 5 max_features: 3 min_samples_leaf: 3 n_estimators: 300	penalty: 'l2'	n_neighbors: 5 weights: 'distance'	learning_rate: 0.1 n_estimators: 200	alpha: 0.05 loss: 'hinge' penalty: 'l2' random_state: 5
Wall Time	10min 14s	4h 12min 20s	6min 46s	16min 8s	2h 4min 40s	1h 28min 21s

V. DISCUSSION

From Table 1, we observed that TF-IDF Vectorizer performed better than Count Vectorizer when used as the vectorizer in most cases. Hence, we selected TF-IDF Vectorizer as our choice vectorizer. TF-IDF Vectorizer focuses on the frequency of words in the corpus as well as their relevance. This allows us to exclude words that are not as significant for analysis and also reduces the input dimension.

Using TF-IDF Vectorizer, we removed stopwords from the data and an improved performance was achieved across all classifiers except KNN. The data was further stemmed and lemmatized. We observed that lemmatization offered better performance in all classifiers when compared to stemming

except in KNN and LR. Table 2 shows the performances when ngram was applied to extract text data to form the vectorizer. Ngram decreases performance for MNB, KNN, and XGB but with little increase in RF, LR, and SGD.

Further performance improvement was gained through hyper parameter tuning as shown in Table 3. It was observed that KNN consistently had the least performance in all cases. With hyper parameter tuning, performance increased a bit but with very high training time in most of the classifiers. It is observed that MNB and SGD had an accuracy of 76% and 74% and a computation speed of 10min 14s and 1h 28min 21s respectively.

VI.CONCLUSION

This paper trained and experimented the performance of six different classifiers (MNB, LR, KNN, RF, XGB, and SGD) in an NPL pipeline. Experimental results showed that MNB Classifier outperformed other state-of-the-art classifiers when its parameter is tuned. The classifier achieved an overall accuracy of 76 % and had the least computation speed except for LR. SGD outperformed all other models when tuned with Using TF-IDF Vectorizer + Stopwords + Lemmatization + Ngram (Unigram+Bigram).

In this study, the results obtained using a single and traditional ML model are relatively low. We suggest that a hybrid model or deep learning approach could yield better performance. In future work, we will experiment using a hybrid classification model for the same task. Research has shown that most hybrid model generally performs better compared to a single model [7-8]. In the hybrid model, we will leverage the advantage of deep learning techniques by finding a classifier that best combines with Convolutional Neural Networks (CNN) which are known to be very efficient in document classification [9-10] to achieve better accuracy.

VII. REFERENCE

[1] Orobor A.I. (2016). Integration and Analysis of Unstructured Datafor Decision Making: Text Analytics Approach. International Journal of Open Information Technologies, 4(10, 82-88

[2] Hassan, S.U., Ahamed, J. & Ahmad, K. (2022). Analytics of Machine Learning-Based Algorithms For Text Classification. Sustainable Operations and Computers, 3, 238–248

[3] Luo, X. (2021). Efficient English text classification using Selected Machine Learning Techniques. Alexandria Engineering Journal, 60, 3401–3409

[4] Gui, Y., Gao, Z., Li, R., & Yang, X. (2012). Hierarchical Text Classification for News Articles Based-on Named Entities. In: Zhou, S., Zhang, S., Karypis, G. (eds) Advanced Data Mining and Applications. ADMA 2012. Lecture Notes in Computer Science, 7713, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-35527-1_27

[5] Rabbimov, M. & Kobilov, S.S. (2020). Multi-Class Text Classification of Uzbek News Articles using Machine Learning. Journal of Physics: Conference Series, in IV International Scientific and Technical Conference, 1546

[6] Scikit-learn 5.6.2. The 20 Newsgroups Text Dataset, 2022. Available at: https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html. (Accessed: 12 May 2022)

[7] Fan, X., Lung, C., & Ajila, S. (2018). Using Hybrid and Diversity-Based Adaptive Ensemble Method for Binary Classification. International Journal of Intelligence Science, 8, 43-74

[8] Dang, C.N., Moreno-García, M.N., & Prieta, F.D. (2021). Hybrid Deep Learning Models for Sentiment Analysis," Complexity, 2021, 1-16

[9] Albawi, S., Mohammed T.A., & Al-Zawi, S. (2017). Understanding of a Convolutional Neural Network. International Conference on Engineering and Technology (ICET), 1-6

[10] Gultepe, E., Kamkarhaghighi, M., & Makrehchi, M. (2018). Latent Semantic Analysis Boosted Convolutional Neural Networks for Document Classification, 5th International Conference on Behavioral, Economic, and Socio-Cultural Computing (BESC), 93-98

[11] Hapke, H. & Nelson C. (2020). Building Machine Learning Pipelines. O'Reilly Media, Inc.

[12] Kim, S.W. & Gil, J.M. (2019). Research Paper Classification Systems Based on TF-IDF and LDA Schemes. Hum. Cent. Comput. Inf. Sci. 9, 30. https://doi.org/10.1186/s13673-019-0192-7

[13] Mustafa, G., Usman, M., Yu, L., Afzal, M.T., Sulaiman, M. & Shahid, A. (2021). Multi-label Classification Of Research Articles Using Word2Vec and Identification of Similarity Threshold. Sci Rep, 11, 21900. https://doi.org/10.1038/s41598-021-01460-7

[14] Harish, B.S. & Udayasri, B. (2014). Document Classification: An Approach Using Feature Clustering. In: Thampi, S., Abraham, A., Pal, S., Rodriguez, J. (eds) Recent Advances in Intelligent Informatics. Advances in Intelligent Systems and Computing, 235. Springer, Cham. https://doi.org/10.1007/978-3-319-01778-5_17

[15] Aphinyanaphongs, Y., Tsamardinos, I., Statnikov, A., Hardin, D., & Aliferis, C.F. (2005). Text categorization models for high-quality article retrieval in internal medicine. Journal of the American Medical Informatics Association : JAMIA, 12(2), 207–216. https://doi.org/10.1197/jamia.M1641