Introduction to Text Classification: Impact of Stemming and Comparing TF-IDF and Count Vectorization as Feature Extraction Technique

André Wendland[1](✉), Marco Zenere[1], and Jörg Niemann[2]

[1] Faculty of Computer Science, Free University of Bozen Bolzano, Piazza Università, 1, 39100 Bolzano, BZ, Italy
[2] Department of Mechanical and Process Engineering, University of Applied Sciences Düsseldorf, Münsterstraße 156, 40476 Düsseldorf, Germany
joerg.niemann@hs-duesseldorf.de

Abstract. Natural language processing is a widely used application in research and industry. Amongst other, use cases are sentiment analysis, speech recognition, classification, query answering and machine translation. In this research we investigate widely applied preprocessing methods, to improve the results of different Algorithms trained on a Fake News data set. As feature extraction methods we compared TF-IDF and Count-Vectorization. TF-IDF yielded slightly better results in terms of accuracy. We found that, as opposed to current research, stemming leads to a minor increase of false positive and false negative classifications, hence to a decrease in accuracy. Among the compared models, logistic regression and support vector machine yielded the best results.

Keywords: NLP · Text classification · Machine learning · Supervised learning

1 Introduction

This paper compares different algorithms for text classification and aims to find optimal preprocessing methods to achieve high accuracy, precision and recall. Based on this findings, further work will focus on an adaptable model which allows to classify text and perform sentiment analysis in an additional building block. Adaptable in the sense that further building blocks of the model can be trained to identify for example fake reviews, which might be interesting for a company that sells goods or services to customers. Linking this to one of the core principles of the SPI manifesto which is to constantly "support the organization's vision and objectives […]" [1] and the fact that in a competitive environment, one of the most important factors in gaining a customer over the competition is reputation, opens a wide range of applications to strengthen a company's market positioning. The classification results can be used to put effort into deleting fake reviews or optimizing flaws in a product or service that are identified in real reviews. This study utilizes a fake news dataset that is labelled and publicly available to demonstrate the feasibility of the first model phase. Figure 1 visualizes a possible use case for a company.

M. Yilmaz et al. (Eds.): EuroSPI 2021, CCIS 1442, pp. 289–300, 2021.
https://doi.org/10.1007/978-3-030-85521-5_19

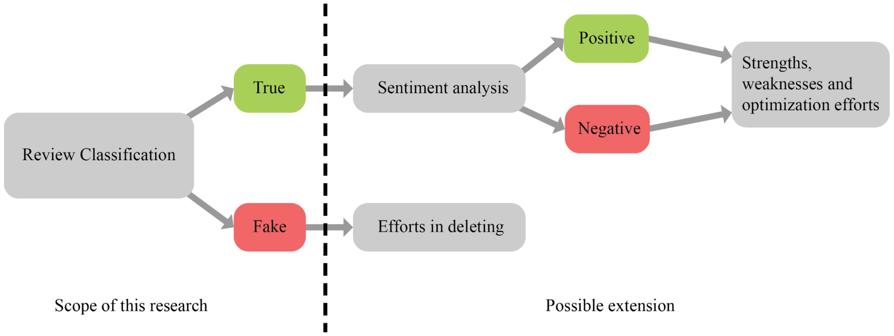


Fig. 1. Proposed use case and extension after classification

1.1 Dataset and Background

The phenomenon of fake news has been known for centuries. It has been abused for propaganda purposes or to justify political decisions. The impact and especially the operating range of fake news increased recently, since significantly more people can be reached through the internet. A current example for the massive impact of fake news or biased information are the elections in America, dating back to 2016. A. Bovet and H. Makse show in their research that among 30,2 million tweets collected within a period of five months before election day, 25% are fake or extremely biased [2]. As seen from the Cambridge Analytica scandal, usage of personalized data and spread of fake or biased news can highly influence the opinion of voters [3]. Since false information can have negative influence on readers, it should be detected as soon as possible. Bringing it into a business context, in 1993, when the internet was still in its infants, Goodman discovered that management decisions are "[…] often based on incomplete or inaccurate information […]" [4], which can lead to severe consequences. Nowadays, biased, fake or wrong information circulate the internet frequently. Basing a strategic decision on fake news, which contrast the real situation, are likely to result in negative consequences. The proposed model in Fig. 1 can therefore be used in supporting the strategic planning at strategy level.

For this reason, the following work focuses on the classification of fake news. A dataset from Kaggle [5] containing about 17900 fake and 20800 true news is chosen to apply different machine learning (ML) algorithms. This work focuses on plain text classification. Metadata, such as sources, publishing dates, media type (print, electronic), are not considered. The goal of this research is to compare the proposed ML algorithms based on specific metrics. For the investigated dataset we can confirm that the chosen machine learning algorithms are able to differentiate between fake and true news. Especially logistic regression and support vector machine achieve good results. Two different feature extraction methods are compared and the impact on the evaluation metrics, when applying stemming, shall be outlined.

2 Methodology and Dataset

The dataset is composed of true and fake news from https://www.politifact.com. The structure of the data can be seen in Table 1.

Table 1. Dataset structure

Title	Text	Subject	Date
Example Title	[source] Example Text	Example Subject	14.04.2021

Each news has as features the title of the article, text, subject and the date at which the article was published. Furthermore, most of the true news come from the same source (Reuters). To avoid learning a source bias, the source has been removed for all samples. Figure 2 shows the applied methodology of this work.

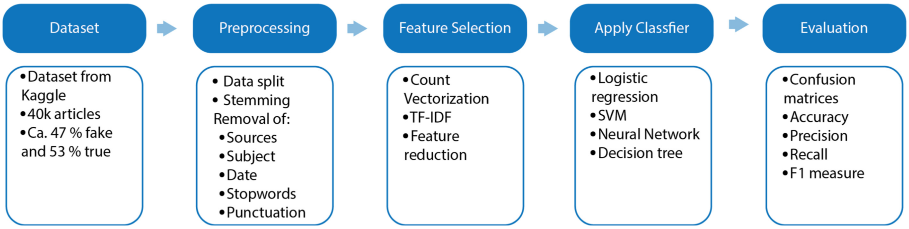


Fig. 2. Methodology

For text classification we are mainly interested in text features. As mentioned in the introduction, the focus will be on the plain text. The first step of preprocessing the data includes transforming all letters to lower case. This simplifies further processing, such as counting word occurrences. After deleting the subject and date columns and merging title and text, the next step is to remove stop words (“a”, “and”, “the”..) from the datasets. They are generally equally distributed through every article and do not give any information about neither the semantics nor the target class. This holds true for punctuation, which has been entirely removed from the datasets. According to [6] stemming helps adding semantic value when it comes to feature selection, “[…] stemming is applied in order to bring words from their current variation to their original root […]”. The results of applying stemming to a set of words from the dataset can be obtained from Table 2.

Table 2. Stemming results

Sample words from the dataset	After stemming
Talking	Talk
Disappointments	Disappoint
Ridiculing	Ridicul
Disappointing	Disappoint
Talked	Talk

One can see that words with a similar semantic meaning, such as “talking” and “talked”, are reduced to “talk”. This helps scaling the number of words to be considered by the models. PorterStemmer from the nltk python library is used within this research. The impact of the stemming process will be discussed in the last section.

2.1 Feature Selection

The following classifiers are implemented in order to distinguish fake from real news:

1. Logistic regression
2. Support Vector Machine (SVM)
3. Neural Network (NN)
4. Decision Tree

They require numerical features as input data, therefore the text of the news needs to be converted. The principle is to convert the text into a vector of word occurrences. This paper focuses on two well studied techniques:

1. Count vectorization
2. Term Frequency Inverse Document Frequency (TF-IDF)

Count vectorization simply counts the number of occurrences a word appears in the document. This results in a bias towards the most frequent words. A disadvantage using this method is that the weight of a rare word is low. In some cases a rare word can have a high semantic importance.

TF-IDF is similar to count vectorization but it introduces a weight factor. It consists of two parts, the Term Frequency (TF) and the Inverse Document Frequency (IDF). TF considers the total occurrence of a word within a document [7]. Since the size of the different articles varies, it might be the case that a certain word occurs more often because the article is longer. To address this, the total occurrence is divided by the total number of words within the article. A simple example: Article a1, part of a set A containing 10 articles, consists in total of 1000 words. The word “Trump” occurs 100 times, then:

$$\mathrm{TF}(\mathrm{Trump}, \mathrm{a1}) = \frac{\text{Total Occurence}}{\text{Total Number of Words}} = \frac{100}{1000} = 0,1 \tag{1}$$

TF treats all keywords equally, regardless the meaning or semantic importance. To emphasize the semantic meaning of a word, IDF is introduced. It considers the frequency among the articles. A higher value will be assigned to terms that occur frequently in a document and are less frequent among all documents. Put into context with the example: the word "Trump" occurs in 5 of 10 articles belonging to A. IDF of Trump among the 10 articles can be calculated as in [6]:

$$\mathrm{IDF}(\mathrm{Trump, A}) = \log_{\mathrm{e}}\left(\frac{|\mathrm{Articles}|}{|\mathrm{Articles\ in\ which\ word\ occurs}|}\right) = \log_{\mathrm{e}}\left(\frac{10}{5}\right) = 0,3 \qquad (2)$$

The TF-IDF parameter of Trump can be calculated as a product of the TF and the IDF values [6]:

$$\mathrm{TF} - \mathrm{IDF}(\mathrm{Trump, A}) = \mathrm{TF}(\mathrm{Trump, a1}) * \mathrm{IDF}(\mathrm{Trump, A}) = 0,03 \qquad (3)$$

The values increase proportionally to the number of times a word appears in the document and is offset by the number of documents that contain the word, which helps to balance that some words appear more frequently in general. We used and compared both approaches to see which of them is more suitable for the chosen dataset. The research community considers TF-IDF as the superior feature extraction method, which we can confirm for the investigated data. As for stemming, which is commonly applied in most NLP projects, we found that it decreased the accuracy.

2.2 Dimensionality Reduction

After preprocessing the dataset, the feature set contains 142 634 unique words. "The standard SVM decision function typically utilizes all the input variables" [8], thus, the "Curse of Dimensionality" could have negative impact on training the models. TF-IDF allows to set a maximum number of words to be considered. Different values for the number of features to be considered have been experimentally evaluated. 30 000 features led to the highest values of the evaluation metrics and will therefore be used to train the models.

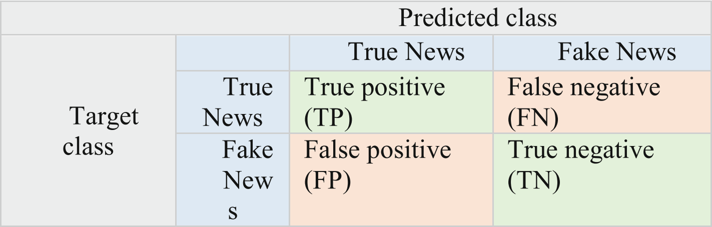
2.3 Evaluation

To evaluate the results of the classifiers, the following metrics are consulted:

- Confusion matrix
- Accuracy
- Precision
- Recall
- F1 score

The dataset is initially split into a test set, consisting of 30% of the whole dataset, and a trainset, containing the remaining 70%. As validation, a 10-fold cross-validation is implemented for each classifier. The python libraries scikit-learn and keras are used to implement classifiers and data split [9, 10]. For the NN, 33% of the trainset are used as validation set.

Figure 3 shows a template of a confusion matrix. Generally, a confusion matrix can consist of many more rows and columns; one column and row for each class. Since the classification of fake news is binary, the example shows only two rows and columns.



		Predicted class	
		True News	Fake News
Target class	True News	True positive (TP)	False negative (FN)
	Fake News	False positive (FP)	True negative (TN)

Fig. 3. Confusion matrix (Color figure online)

The green diagonal shows the correctly predicted classes. In the case of fake news classification, true positive means a true article has been correctly classified as a true article, respectively, true negative means that a fake article has been correctly classified as fake news. The red diagonal, on the other hand, shows the number of articles that were classified wrong. False positive for the proposed case means a fake news was classified as true news and false negative, correspondingly, a true news was classified as fake news. The matrix consists of absolute numbers and each field will show the total amount of correctly / wrongly classified articles.

The **accuracy** is probably the most widely used metric to evaluate a classification model. It can be seen as the overall recognition rate of the classifier and can be calculated as shown in Eq. 4.

$$\text{Accuracy} = \frac{\text{Amount of correct classifications}}{\text{Amount of samples}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4}$$

The **precision** can be calculated as shown in Eq. 5:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

This metric shows the ratio between correctly classified true news and the total amount of classified true news, thus containing fake news that were classified as true news.

The **Recall** can be calculated as shown in Eq. 6:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

Recall for the proposed study gives information about the ratio of correctly classified true news to the total amount of true news within the dataset. Both metrics should not be consulted isolated. It depends on the use case whether a high recall or high precision or both are desirable. A current example are covid-19 tests. A low precision indicates, that within the sample many participants have been tested positive even though they are not infected. A low recall instead indicates that there is a high number of tests that yielded negative results, even though they should have been positive. For this case, a high recall is most desirable. Generally speaking, recall is important when the cost of false negatives is high.

The F1-Measure combines both metrics, precision and recall. The harmonic mean is computed as shown in Eq. 7.

$$\text{F1-Measure} = \frac{2\text{rp}}{\text{r} + \text{p}} \tag{7}$$

3 Classifiers

3.1 Logistic regression (LR)

Both logistic regression and Support Vector Machine (SVM) are supervised machine learning algorithms in the context of machine learning. According to [11] LR is one of the earliest methods for classification. LR comes with multiple benefits, such as probability modelling [11]. Given a binary classification problem, its goal is to predict the probability of a variable being fake (0) or true (1), given the input of an article.

3.2 Support Vector Machine (SVM)

As deeply explained by Joachims in 1998, SVM is a suitable choice for text classification [12]. It is widely used for this application. Since SVM was originally developed for binary classification tasks, the proposed use case is appropriate [11]. The function of SVM is similar to LR. It separates data by a hyperplane. Corresponding to this hyperplane, the data is assigned to different classes. The goal is to separate the data by the highest margin.

3.3 Neural Network (NN)

Shervin et al. showed in their review of deep learning methods for text classification, that feed-forward Neural Networks belong to the simplest deep learning models for this task. Yet, many implementations have shown high accuracy on text classification benchmarks [13]. Therefore, we chose to implement a simple feed forward network. The architecture is shown in Fig. 4:

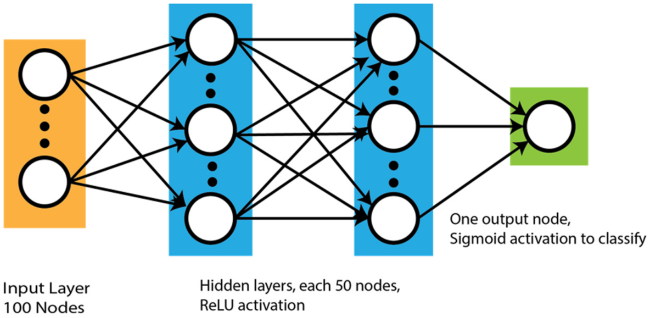


Fig. 4. Neural network architecture

The network receives as input a vector produced by applying TF-IDF/Count vectorization. This input is used in order to classify whether the specific article is fake or not [14]. The input layer consists of 100 neurons, the hidden layers of 50 each and the output layer comes with one neuron which performs the binary classification task (fake or true news). The hidden layers apply rectified linear units (ReLUs) as activation function. This activation function addresses the vanishing gradient problem, which leads to poor learning in deep networks. ReLUs "have been found to yield superior results in many different settings" [15].

3.4 Decision Tree

Within the tree structure, the internal nodes test the incoming data and based on the criteria the data flows through a branch. This leads to another node or to a leaf (terminal node). The terminal node represents the class label. The branches can be seen as "[…] conjunctions of features that lead to those class labels […]" [16]. To decide the path for a certain data input, the model learns simple decision rules inferred from the data features.

4 Evaluation – Results and Discussion

The models are compared using TF-IDF as feature extraction technique. We expected that TF-IDF would lead to remarkable better results than count vectorization. The expectation is based on the fact that this method of feature extraction considers both, word frequency and inverse document frequency. The latter considers the semantic importance of words by adding a weight factor as shown in Eq. 2. Table 3 shows the average of total misclassifications of LR and SVM. The values were extracted and averaged from the confusion matrices of the LR and SVM Models of several prediction runs.

Considering that in total 13 470 articles within the test set were classified, the overall accuracy of both, TF-IDF and count vectorization, is high. With count vectorization, LR achieves an accuracy of 98,91% and SVM 98,87%. The result confirms the initial assumption partly. Since the accuracy of count vectorization is very high, the room for improvement is small. Nevertheless, applying TF-IDF yields better results: SVM

Table 3. Absolute number of missclassifications

	LR	SVM
TF-IDF	117	111
Count vectorization	169	195

increases to 99,18% and LR to 99,13%. The research by Poddar et al. deals with fake news detection and they confirm the initial assumption [17].

Table 4 shows the result of the comparison of the different ML models based on the metrics explained in Sect. 2. The values correspond to the average of successive predictions. The first value of Precision and Recall belongs to the prediction of the true news and the second value, respectively, to the prediction of fake news.

Table 4. Comparison of metrics for TF-IDF

Model	Accuracy	F1 score	Precision	Recall
LR	0,9913	0,99	0,99	0,99
			0,99	0,99
SVM	0.9918	0,99	0,99	0,99
			0,99	0,99
NN	0,9866	0,99	0,99	0,98
			0,98	0,99
Decision tree	0,9631	0,96	0,96	0,97
			0,97	0,96

One can see that the decision tree, achieving an overall accuracy of 96,31%, performs rather poor. This was also observed in [17]. SVM performed marginally better than LR. The NN has slightly lower accuracy and recall than the LR and SVM. The best Precision and Recall values are achieved by LR and SVM, followed by the NN with a deviation of 1% in the recall. The decision tree shows, with up to 3%, the highest deviation to the best observed metrics. Other publications, for example [18], show similar metric scores for NN and SVM (between 99,80 and 99,9%) applied on twitter tweets.

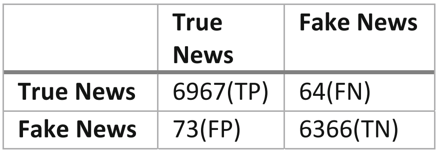
The following evaluations will concentrate on LR and SVM because they produced the best results. Most of the conducted studies on fake news classification show a pattern; they implemented or recommend word stemming or lemmatization [6, 11, 12, 19]. Therefore, we compared the metrics before and after implementing stemming, using PorterStemmer from the python library nltk. We found that the accuracy decreased, however, not remarkably as shown in Table 5.

Over successive predictions, there was a slight decrease in accuracy and, as a result, a slight increase in total misclassifications. We consult the confusion matrices to help explain this (Table 6):

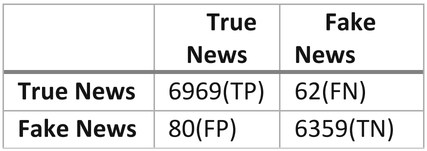
Table 5. Accuracy with and without stemming

	LR	SVM
Without stemming	0.9898	0.9881
Applying PorterStemmer	0.9894	0.9869

Table 6. LR before stemming (left) and LR after stemming (right).



	True News	Fake News
True News	6967(TP)	64(FN)
Fake News	73(FP)	6366(TN)



	True News	Fake News
True News	6969(TP)	62(FN)
Fake News	80(FP)	6359(TN)

As seen from the matrices, the stemming operation leads to a higher number of true positives (TP) and a higher number of false positives (FP). The same can be observed for the SVM confusion matrices. Since the Accuracy is dominated[1] by Precision it decreases slightly. According to published studies, stemming reduces dimensionality without sacrificing accuracy [20]. This is not the case for the processed dataset, as seen above. Table 2 showed a sample result of the stemming process applied on words of the dataset. To further elaborate on a negative impact of stemming we consider the following words: operations and operative. By applying PorterStemmer both words are stemmed to “oper”. The original words can be used in different context, for example in “business operations” and “operating system”. In this case, stemming leads to a semantic loss which is embedded in the vectorization (TF-IDF or count vectorization) and hence impacts the training of the model.

4.1 Hyperparameter Tuning

The results indicate that LR and SVM are the most suitable choices for the analyzed dataset. Therefore, we tested different values for the regularization rate λ. Regularization is the penalty of a model’s complexity and helps adjusting over- and underfitting [20]. When using TF-IDF as extraction technique, we found that $\lambda = 20$ and the maximum number of iterations = 200 led to the best results. Accuracy is increased by 0,3% in comparison to the standard settings (λ= 1 max. iterations = 100). SVM on the other hand, yields the best results when maintaining the standard settings, $\lambda = 1$, max. iterations = 1000.

5 Limitations, Conclusion and Further Work

The data is heterogenous, which is one of the limitations of fake news classification. This is due to the vast amount of possible sources of fake news. These include social media,

[1] The dataset is not completely balanced and contains more true news than fake news.

news reporting, print media, and other forms of media that may have some parallels but differ significantly. Therefore, the deployed models within this research can have anomalies and might not generalize well on other data.

When considering preprocessing steps, many studies implement or recommend word stemming/lemmatization as an easy way to reduce dimensionality without degrading accuracy. For the processed dataset and the chosen models this does not hold true, possibly due to semantic loss applying stemming. We found that TF-IDF as feature extraction technique yields slightly better results than count vectorization for the given data. This can be explained based on the assumption that the semantics of less occurring words within the corpus are considered more valuable. However, as shown for stemming, this must not always be the case.

Further work will include validating the findings on other datasets and implementing a complete pipeline as proposed in Fig. 1. To evaluate the impact of stemming in more detail, further data sets will be investigated. If a negative impact is found, the most occurring stems within the vectorization will be extracted and put into context with the original data. Following this approach, an evaluation of a possible semantic loss is possible.

References

1. SPI manifesto. https://2020.eurospi.net/index.php/manifesto. Accessed 01 Feb 2021
2. Bovet, A., Makse, H.A.: Influence of fake news in Twitter during the 2016 US presidential election. Nat Commun **10**(1), 7 (2019). https://doi.org/10.1038/s41467-018-07761-2
3. Boldyreva, E.L.: Cambridge analytica: ethics and online manipulation with decision-making process, pp. 91–102, December 2018. https://doi.org/10.15405/epsbs.2018.12.02.10
4. Goodman, S.K.: Information needs for management decision-making. ARMA Rec. Manage. Q. **27**(4), 12 (1993)
5. Fake and real news dataset. https://kaggle.com/clmentbisaillon/fake-and-real-news-dataset. Accessed 05 May 2020
6. Biba, M., Gjati, E.: Boosting Text Classification through Stemming of Composite Words. In: Thampi, S.M., Abraham, A., Pal, S.K., Rodriguez, J.M.C. (eds.) Recent Advances in Intelligent Informatics, vol. 235, pp. 185–194. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-01778-5_19
7. Hakim, A.A., Erwin, A., Eng, K.I., Galinium, M., Muliady, W.: Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency (TF-IDF) approach. In; 2014 6th International Conference on Information Technology and Electrical Engineering (ICITEE), Yogyakarta, Indonesia, pp. 1–4, October 2014. https://doi.org/10.1109/ICITEED.2014.7007894
8. Dasgupta, S., Goldberg, Y., Kosorok, M.: Feature elimination in kernel machines in moderately high dimensions. arXiv:1304.5245*[stat]*, December 2015. http://arxiv.org/abs/1304.5245. Accessed 07 May 2020
9. Scikit-learn: machine learning in Python—scikit-learn 0.22.2 documentation. https://scikit-learn.org/stable/. Accessed 12 May 2020
10. Keras: the Python deep learning API. https://keras.io/. Accessed 12 May 2020
11. Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., Brown, D.: Text classification algorithms: a survey. Information **10**(4), 150 (2019). https://doi.org/10.3390/info10040150

12. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0026683
13. Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., Gao, J.: Deep learning based text classification: a comprehensive review. arXiv:2004.03705*[cs, stat]*, April 2020. http://arxiv.org/abs/2004.03705. Accessed 12 May 2020
14. A Beginner's Guide to Bag of Words & TF-IDF. Pathmind. http://pathmind.com/wiki/bagofwords-tf-idf. Accessed 12 May 2020
15. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Deep learning. In: Data Mining, pp. 417–466. Elsevier (2017). https://doi.org/10.1016/B978-0-12-804291-5.00010-6
16. Sharma, H., Kumar, S.: A survey on decision tree algorithms of classification in data mining, April 2016. https://www.researchgate.net/publication/324941161_A_Survey_on_Decision_Tree_Algorithms_of_Classification_in_Data_Mining
17. Poddar, K., Amali D, G.B., Umadevi, K.S.: Comparison of various machine learning models for accurate detection of fake news. In: 2019 Innovations in Power and Advanced Computing Technologies (i-PACT), Vellore, India, pp. 1–5, March 2019. https://doi.org/10.1109/i-PACT44901.2019.8960044.
18. Aphiwongsophon, S., Chongstitvatana, P.: Detecting fake news with machine learning method. In: 2018 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Chiang Rai, Thailand, pp. 528–531, July 2018. https://doi.org/10.1109/ECTICon.2018.8620051
19. Ahmed, H., Traore, I., Saad, S.: Detection of online fake news using n-gram analysis and machine learning techniques. In: Traore, I., Woungang, I., Awad, A. (eds.) ISDDC 2017. LNCS, vol. 10618, pp. 127–138. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69155-8_9
20. Regularization for Simplicity: Lambda | Machine Learning Crash Course. https://developers.google.com/machine-learning/crash-course/regularization-for-simplicity/lambda. Accessed 12 May 2020