Filtering Clickbait from News Feeds

Daniel Bae
dbae32@gatech.edu
Submission Date: April 17, 2022
Term / Course: Spring 2022 / ISYE-7406

Abstract

Free news feeds, such as Google News, are broadly available and accessible sources of information [1]. However, the proportion of clickbait articles that comprise these feeds has become hard to ignore. Even reputable news sources have turned to using clickbait tactics. This paper explores the potential ability of interpretable machine learning models to classify articles as either non-clickbait or clickbait based on only their titles, and to describe the characteristics that differentiate the two types of titles. By relying on only article titles, the models avoid learning to associate other attributes of an article that may or may not be linked with either class over time. For example, a reputable news source that avoided using clickbait in the past may lean on clickbait strategies more in the future. If a model were to learn to associate an article's source with its predicted classification, it would require frequent, potentially even live, updates to keep track of such trends. These experiments demonstrate that even relatively simple models, given just article titles and limited examples, can attain non-clickbait versus clickbait classification performance different from random guessing.

1. Introduction

Clickbait has been generally defined as, "something (such as a headline) designed to make readers want to click on a hyperlink especially when the link leads to content of dubious value or interest" [2]. For the specific purposes of this research, clickbait news articles were additionally defined as those with titles that could be interpreted as being advertisements, hyperbolic, intentionally obfuscating, or highly opinionated/biased. Non-clickbait news articles were defined as those with titles that were seemingly straightforward and explicit in stating their contents.

The initial intent of this research was to classify news articles as having positive, negative, or neutral sentiment. However, it quickly became apparent that the prevalence of clickbait would make collecting data and generating consistent labels quite a challenge. This real-world use case, and other potential use cases, became the motivation for learning a data mining process to efficiently separate non-clickbait news articles from clickbait and to be able to effectively describe their differences.

Other studies have attempted, and succeeded to various degrees, to identify misinformation or "fake news" in a variety of contexts using several attributes of, for instance, social media posts, blogs, websites, etc. [3]. An important distinction from these other studies is that this research specifically ventured to demonstrate that the use of manipulative techniques in news article titles could be picked up based on the information present in the titles alone. It does not claim to describe a method for identifying misinformation nor does it describe a method for making any assumptions about a news article's contents based on its title.

The experiments conducted focused on news articles pertaining to organizations specifically as it was uncertain whether the language used to describe individuals and/or concepts could materially differ and what implications that would have for experiment design. To some degree, it should be feasible to extrapolate some of the findings to those other contexts.

Ultimately, it was discovered through experimentation that a Multinomial Naïve Bayes classifier can separate non-clickbait from clickbait. The results were especially promising considering that the training examples collected for the experiments were less than ideal both in terms of quantity and quality. The Multinomial Naïve Bayes model was also able to describe differences between the titles from the two classes in an almost directly human interpretable form.

2. Data

The sample data used for experimentation was acquired from Google News through an RSS feed [1]. A Python script was written to collect the data and class labels. The script prompted a user for a subject, then presented the user with recent news article titles pertaining to said subject one at a time. The user was expected to provide their response to each title by entering either a "1" or "0" from their keyboard which would correspond to labels "non-clickbait" or "clickbait" respectively. In total, 1,196 examples were collected and labelled across 12 subjects. The number of examples per subject was roughly uniformly distributed, meaning there were approximately 100 examples per subject. Clickbait comprised around a quarter of the sample population. This meant that the data set was unbalanced and certain design decisions would have to factor this in.

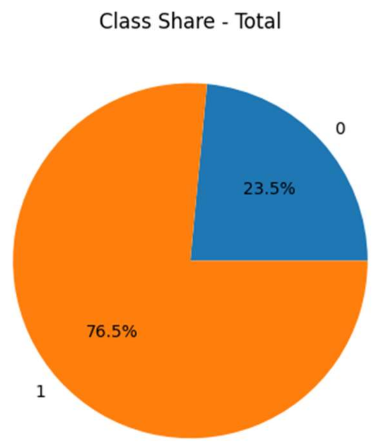


Figure 1: A pie chart showing the proportion of the overall sample data belonging to each class.

All of the examples were labelled by one individual. This decision potentially introduced a large degree of inconsistencies and the individual's personal biases into the data on top of the already expected random noise. Other possible consequences of this design decision are discussed further in the "Conclusions" section.

3. Methods

Several transformations were applied to the raw data in order to get it into a format that could be ingested by a classifier. If the subject name appeared in a title pertaining to said subject, the name was replaced with the arbitrary string "Abcde". This was done to prevent any models from learning to associate a specific subject name with either class. The titles were then lemmatized. Lemmatization refers to the act of reducing a word to its lemma, or base form, while taking context into consideration, to some extent, by referencing the word's part-of-speech (PoS) [4]. For instance, the word "learning" might be lemmatized to "learn". This was done in order to group words with similar meanings into a single feature rather than having multiple features to represent them. The lemmatized titles were then vectorized. Each word or unit in a title was transformed into an individual feature and became part of an overall vocabulary. Vocabulary here refers to every word or unit that was encountered across all of the examples in the data set collected and represents the feature space. Features are referred to as "words" or "units" because a title may contain information other than words such as numbers, dates, punctuation, etc. The value for each of the features then became the number of times a word or unit appeared in each individual title. The size of the vocabulary was approximately 4,000 features. These included common stop words and punctuation.

There does not exist an agreed upon list of stop words, but they are generally considered words that may modify a given context or the meaning of other words, however, have little meaning on their own [5]. Stop words, such as "which", and punctuation are often automatically removed when performing data preparation for natural language processing (NLP) tasks. For the purposes of this research, stop words were not imported from an existing list and punctuation was not explicitly removed. The reason for this being that, words such as "which" and punctuation such as "?" often appear in clickbait and may provide valuable information for discerning non-clickbait from clickbait. Take for instance a common clickbait title, "Which company just announced big news?!". Instead, stop words were "learned" as being words or units that appeared in more than 97% or fewer than 3% of the overall sample of news article titles. This reduced the feature space from over 4,000 to just 44. A correlation matrix of these 44 features revealed that, as a whole, there was little correlation among them. Multicollinearity should therefore have had little to no influence and any assumptions of independence made were likely not too overreaching. Unfortunately, but perhaps reasonably, none of the features exhibited strong correlation with the response either, suggesting that no single word or unit alone is enough evidence to classify a title.

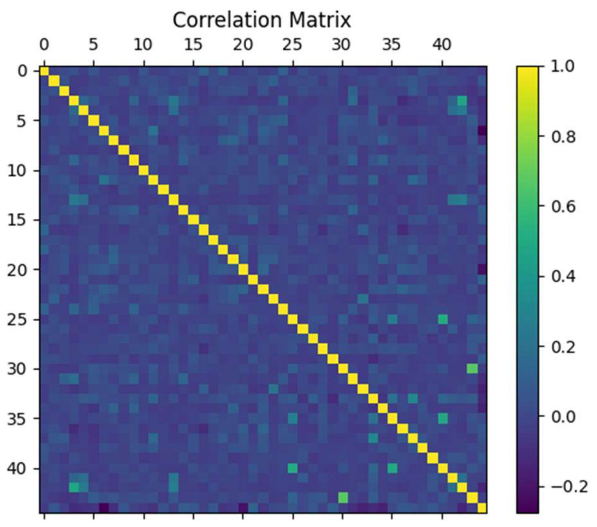


Figure 2: Correlation matrix of the features among themselves and with the response (the final row/column).

The two models selected for examination were the Random Forest (RF) and Multinomial Naïve Bayes (MNB) classifiers. The models were built, and ancillary functions were leveraged, through the sklearn Python package [6]. Both of these models fulfilled the requirement of providing predictions while being “interpretable” but with two very different approaches. The RF classifier is an ensemble method that takes the popular vote across many trees trained in parallel in order to provide predictions that are generally more robust than those provided by any individual tree [7]. At a high level, each tree attempts to successively split the data on some feature and some value for said feature until some termination criteria is met. The general goal is to maximize information gain or minimize uncertainty. RF classifiers take advantage of bootstrap aggregation (bagging) in order to reduce variance in prediction by randomly sampling the training data with replacement. They also perform feature selection inherently and are agnostic to data scales. For these reasons, and many more, RF classifiers were considered some of the most generally applicable and capable models, before the advent of deep learning techniques, and were an obvious choice for consideration. MNB classifiers on the other hand are simple probabilistic models that take advantage of Bayes’ theorem in order to provide surprisingly reliable results even though they make a strong independence assumption [8]. They have been a popular choice for text classification tasks, for example email spam detection, even though their independence assumption rarely holds in those scenarios. Several other approaches were considered, such as distance-based methods like k-nearest neighbors (KNN) or support-vector machines (SVM), but were ruled out for failing to meet certain criteria. For example, KNN is a nonparametric method and instead of “learning” can be better characterized as “memorizing” relationships [9]. This means that it does not scale well as the number of training examples and/or features grows large. It requires all of the training data to be kept in memory in order to make predictions and does not provide easily interpretable or intuitive results. Other models were eliminated following a similar evaluation process.

The two selected models were then compared to a random “classifier”. Here the word “classifier” is used a bit loosely. A random number generator would give “predictions” of either “1” or “0” and was designed to be three times as likely to produce a “1” compared to a “0” as opposed to having equal probability for either output. This was done in order to better simulate the observed proportions in the sample population (see figure 1). The random classifier would serve as a control to compare against. If the two selected models performed similarly to the random classifier, this would indicate that they were no better at separating non-clickbait from clickbait than random guessing and that they offered little value.

The total data set was randomly split into training and testing partitions. The training data set comprised 70% of the original total data set and the remaining 30% was reserved for testing. The training and testing data sets were carefully shuffled in order to conserve the original proportions of the total dataset (see figure 1) within 0.1% so that any comparisons made or conclusions drawn would remain like-for-like. The hyperparameters of the two selected models were tuned using the training data set. A range of parameter values for each model was searched following an exhaustive grid search procedure, applying 3-fold cross-validation, and using F1-score as the evaluation metric. F1-score is a metric that combines the fraction of true positives over all labelled positives (precision) and the fraction of true positives over all actual positives (recall) and it is often a more appropriate performance measure than simple accuracy when a class imbalance is present [10].

$$F_1 = \frac{2}{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} = \frac{\mathrm{tp}}{\mathrm{tp} + \frac{1}{2}(\mathrm{fp} + \mathrm{fn})}$$

The random guesser did not require any hyperparameter tuning. Once the local ideal set of hyperparameters, based on best F1-score, had been found for the RF and MNB models, they were then compared to the random guesser. Monte Carlo cross-validation (MCCV) was leveraged to better make use of limited training/validation data and to collect more rigorous aggregate statistics than simple K-fold cross-validation or leave-one-out cross-validation (LOOCV) [11]. The training/validation data set (which comprised 70% of the original total data set) was further randomly partitioned into validation training and validation testing sets comprising 70% and 30% of the data respectively. Then each of the two models and the random guesser were trained on the validation training data and tested against the validation testing data. For the random guesser, “training” meant calculating the empirical class distribution of the validation training data. This procedure was repeated 100 times. The F1-scores were recorded for each of the models at every iteration. Using this data, the mean and standard deviation of each model’s performance was calculated. Finally, the models were evaluated against the testing data set that had been originally reserved and which none of the models had encountered thus far.

4. Results

The local optimal RF and MNB classifiers produced extremely similar MCCV results while the random guesser performed the worst (model hyperparameters can be found in appendix 6.1). The RF classifier did technically exhibit the best MCCV performance. It had slightly higher mean F1-score and slightly lower standard deviation than the MNB model.

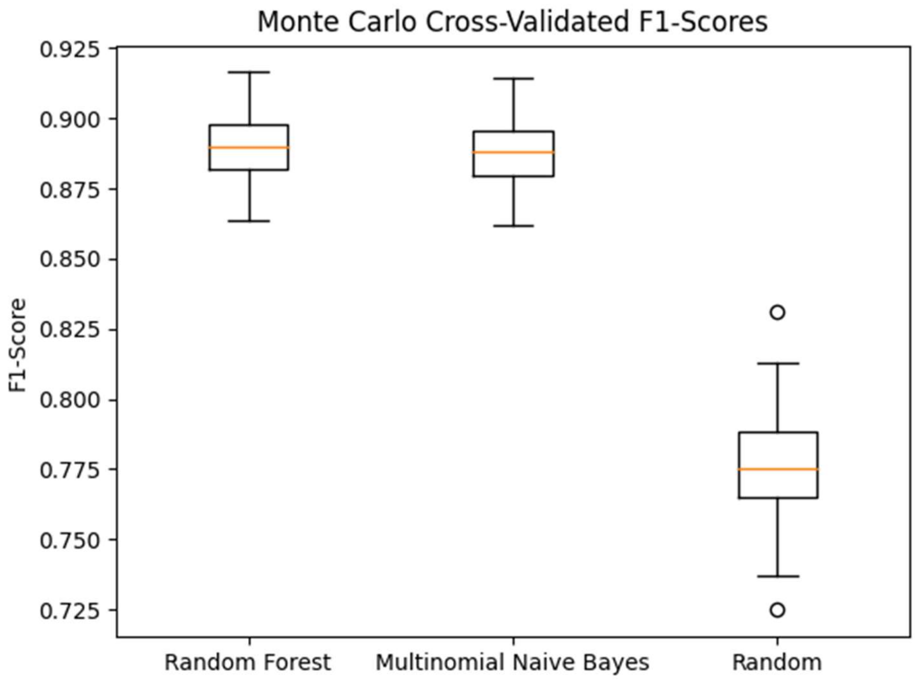


Figure 3: Boxplot representing the 100 iteration MCCV results. A table containing the results in numeric form can be found in appendix 6.2.

The final test, performed on the testing data set that was reserved, generally followed the same patterns but with a few caveats. Overall, the RF classifier again performed the best in terms of F1-score, MNB came in a close second, and the random guesser performed the worst. However, upon inspecting the precision and recall, it was discovered that while MNB had worse precision, it did have slightly better recall than the RF classifier. All three models struggled with precision and seemed to hit a F1-score plateau at around 90%.

n=1, train=837, test=359	F1-score	Precision	Recall
Random Forest	89.08%	82.81%	96.36%
Multinomial Naïve Bayes	88.70%	81.65%	97.09%
Random	78.35%	77.11%	79.63%

Table 1: Table showing the final test results.

Both the RF and MNB methods were able to present information on feature importance in intuitive and human interpretable formats. The RF model returned mean decrease in impurity (MDI) overall, while the MNB model returned log probabilities for each of the features per class. The log probabilities were converted to odds ratios, using the following formulae, for further ease of interpretation.

$$Odds_0 = P(1|x)/(P(0|x) * (P(0)/P(1)))$$
$$Odds_1 = P(0|x)/(P(1|x) * (P(0)/P(1)))$$

The top ten features in terms of MDI as well as the top ten features per class in terms of odds ratios shared a lot of overlap. For example, the RF model suggested that the presence of question marks is the single most predictive indicator in terms of classifying a title as non-clickbait or clickbait. Similarly, the MNB model learned that question marks are around 5 times as likely to appear in clickbait titles than non-clickbait titles (charts providing more details on the feature importance findings can be found in appendix 6.3).

5. Conclusions

The boxplot of the MCCV F1-scores (figure 3) provides evidence that the RF and MNB models were indeed able to achieve performance on these experiments different from random guessing. Though the evidence is not a replacement for hypothesis testing, which would require many more independent trials, there was no overlap at all between the RF and MNB results with the results produced by the random guesser. That being said, the models did struggle with accurately identifying clickbait. They were better at identifying non-clickbait accurately but had relatively high false positive and low true negative rates (see confusion matrices of test results in appendix 6.4). This likely contributed to the F1-score plateau that they seemed to hit at around 90%. Some contributing factors may have included the deficit in quantity and deficient quality of the sample data available. Having a single labeler meant that a relatively small number of examples were available for the experiments. The labels were also highly subject to the individual's biases and whims. There were likely many instances where the individual chose to label an example as belonging to one class and another similar example later as belonging to the other class. In addition, some of the titles may have fallen into gray areas that were difficult to classify. Examples that posed a challenge for a human classifier would likely pose an even greater challenge for a machine. Potential design improvements that may have mitigated these effects include having multiple labelers labelling the same examples, taking the fraction of positive labels over negative labels as the final label, and collecting many more examples. Instead of building a hard classifier, this new type of data would allow a soft classifier to be built that would predict confidence that an example belonged to either class as a percentage.

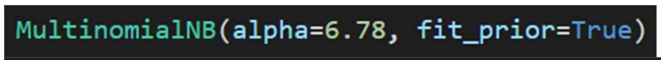
Inherent characteristics of the RF and MNB models may have also contributed to the performance ceiling that was observed. The MNB classifier makes a strong independence assumption. In the context of text analysis, this means that the MNB classifier assumes that each word or unit in a string has no relation to any other words or units. This is clearly false since individual words/units in human language are combined to form phrases or sentences with holistic meanings. The RF model also lacks the ability to learn the structure of phrases or sentences though it does not make a strong independence assumption like the MNB model does. Modern deep learning methods have had much more success in learning to understand human language more like humans do. A combination of many techniques have made this a possibility. For example, recurrent neural networks can have structures such as feedback loops that allow them to accept inputs of variable length and learn the composition of sequential data making them powerful tools for understanding language [12]. Other NLP techniques such as Word2vec make use of neural networks to learn word associations based on large volumes of examples of written human language [13]. The increased complexity may mean that more effort is required in interpreting the results but these are potential avenues for exploration and further research for the purposes outlined in this paper.

Human behavior, such as human language, is notoriously difficult to predict or simulate. Given the circumstances, even though there was plenty of room for improvement, the selected models performed impressively well. Between the RF and MNB models, though the RF model did achieve slightly better performance, several other factors made the MNB model more attractive. The MNB model performed nearly just as well while being much simpler. The hyperparameter tuning times for the RF model was measured in minutes while it was measured in just seconds for the MNB model. Though the MNB model's feature importance results required some manipulation, they were well worth it. The odds ratios have a more direct translation to layman's terms than MDI and were provided per class as opposed to overall. For these reasons, the MNB model better satisfies the objective of the research to discover a method for differentiating non-clickbait from clickbait at an observable level different from random guessing while also revealing interpretable insights.

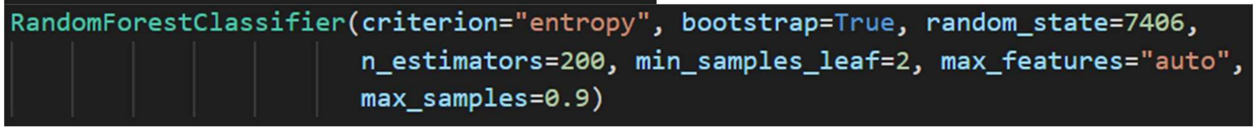
6. Appendix

6.1. Model hyperparameters

These were the locally optimal hyperparameters found for each of the models using exhaustive grid search.



```
MultinomialNB(alpha=6.78, fit_prior=True)
```



```
RandomForestClassifier(criterion="entropy", bootstrap=True, random_state=7406,
                       n_estimators=200, min_samples_leaf=2, max_features="auto",
                       max_samples=0.9)
```

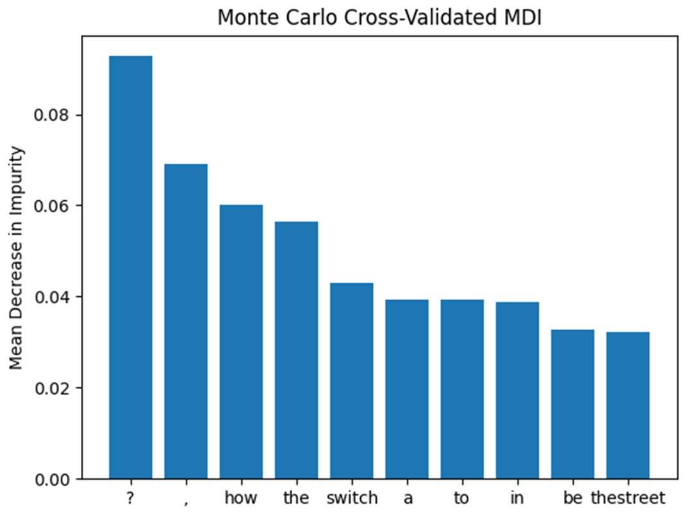
6.2. Table of MCCV results

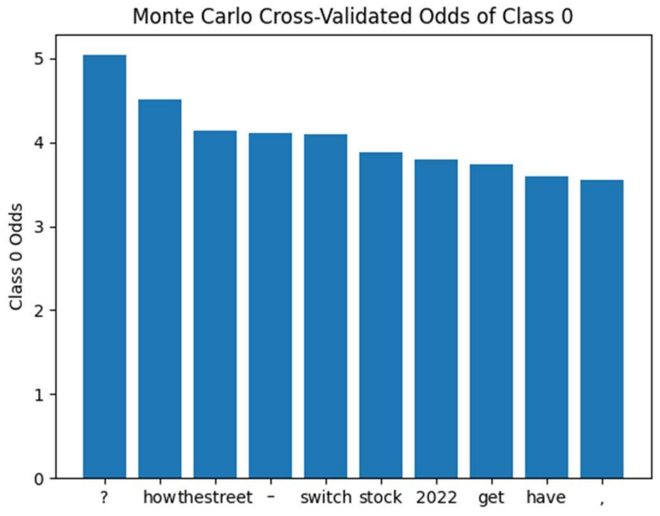
These were the results of the 100 iteration MCCV that supplied the data for figure 3.

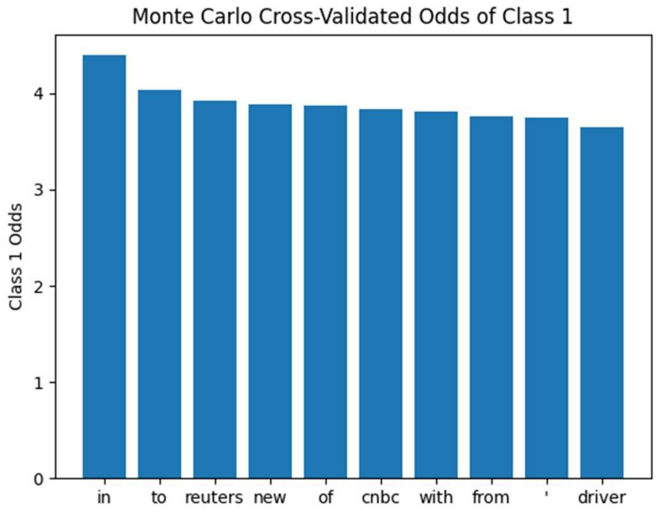
n=100, train=586, test=251	Average F1-score	Average Stdev
Random Forest	88.94%	0.0115
Multinomial Naïve Bayes	88.76%	0.0119
Random	77.68%	0.0178

6.3. Feature importance

These charts show the top ten features in terms of importance by MDI or odds ratios. Source names such as “thestreet” and “reuters” were inadverdently picked up as features because they often appear in the titles. These should likely be stripped in the future for reasons mentioned in this paper.







6.4. Test confusion matrices

These are the results of the final test evaluation in the form of confusion matrices.

Random Forest	Actual=1	Actual=0
Prediction=1	265	55
Prediction=0	10	29

Multinomial NB	Actual=1	Actual=0
Prediction=1	267	60
Prediction=0	8	24

Random Guesser	Actual=1	Actual=0
Prediction=1	219	65
Prediction=0	56	19

References

[1] Google News. [Online]. Available: https://news.google.com/?hl=en-US&gl=US&ceid=US%3Aen. [Accessed: 16-Apr-2022].

[2] “Clickbait definition & meaning,” *Merriam-Webster*. [Online]. Available: https://www.merriam- webster.com/dictionary/clickbait. [Accessed: 15-Apr-2022].

[3] S. Hakak, M. Alazab, S. Khan, T. R. Gadekallu, P. K. Maddikunta, and W. Z. Khan, “An ensemble machine learning approach through effective feature extraction to classify fake news,” Future Generation Computer Systems, vol. 117, pp. 47–58, 2021.

[4] Stemming and lemmatization. [Online]. Available: https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html. [Accessed: 16-Apr-2022].

[5] “Stop word,” Wikipedia, 27-Nov-2021. [Online]. Available: https://en.wikipedia.org/wiki/Stop_word. [Accessed: 16-Apr-2022].

[6] Pedregosa, F. et al., 2011. Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), pp.2825–2830.

[7] “Random Forest,” Wikipedia, 01-Mar-2022. [Online]. Available: https://en.wikipedia.org/wiki/Random_forest. [Accessed: 16-Apr-2022].

[8] “Naive Bayes classifier,” Wikipedia, 04-Mar-2022. [Online]. Available: https://en.wikipedia.org/wiki/Naive_Bayes_classifier. [Accessed: 16-Apr-2022].

[9] “K-nearest neighbors algorithm,” Wikipedia, 15-Mar-2022. [Online]. Available: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm. [Accessed: 16-Apr-2022].

[10] “F-score,” Wikipedia, 18-Mar-2022. [Online]. Available: https://en.wikipedia.org/wiki/F-score. [Accessed: 17-Apr-2022].

[11] Q.-S. Xu and Y.-Z. Liang, “Monte Carlo Cross validation,” Chemometrics and Intelligent Laboratory Systems, vol. 56, no. 1, pp. 1–11, 2001.

[12] “Recurrent neural network,” Wikipedia, 17-Apr-2022. [Online]. Available: https://en.wikipedia.org/wiki/Recurrent_neural_network. [Accessed: 17-Apr-2022].

[13] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J., 2013. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 26.