Think This a Normal Research Paper? Wait Till You See What's Inside! – Feature Selection For Clickbait Detection

Sho Cremers
s.a.cremers@student.tudelft.nl
(5052602)

Sukhleen Kaur
s.kaur@student.tudelft.nl
(5053307)

Kanya Paramita Koesomo
KanyaParamitaKoesoemo@student.tudelft.nl
(5035597)

ABSTRACT

Another day, another headline desperately trying to get your attention and waste your time. These headlines, known as clickbaits, tend to build up curiosity in the user only to leave them disappointed and unfulfilled with what they just witnessed. That is why this study developed an automatic detection of clickbait so that users can be warned of such headlines. To do so, features such as sentence structure, words, phrases, POS-tagging and sentiment analysis are used. Certain features are dropped based on feature importance generated using XGBoost, and three classifiers — SVM, Random Forest and XGBoost — are trained on the selected features. The results indicate that the classifiers are able to perform well with their AUCs and F1-scores ranging between 0.85 to 0.87. In conclusion, this study was successfully able to detect clickbaits rather well.

KEYWORDS

Clickbait, POS-tagging, Sentiment analysis, Feature importance, SVM, Random Forest, XGBoost, AUC, F1-score

1 INTRODUCTION & BACKGROUND

In recent decades, online news media has been one of the main sources of information used to inform people. With the fast growth of information on the internet, information providers need to put in a lot of effort to attract readers' attention. Most online media revenues rely heavily on the click-through rate of their online articles. Thus, with the nature of information growth on the internet and also the increasing number of online news provider competitors, they need to be able to get readers' attention faster than the others. One of the most popular ways of doing so is creating clickbaits, which are headlines that provide "forward referencing" signals to generate enough interest in the readers such that they click on the link [3]. Clickbait headlines usually function as teasers to the reader and they tend to make it sounds more important than it is [7]. Even if a reader clicks on one of these clickbaits, they are not satisfied with the content as there is a gap between what the headline makes the article seem to be from the actual content [3]. As a consequence, clickbait tends to waste the readers' time and it develops a sense of being tricked as it does not meet their expectations.

In this project, we aim to develop an automatic detection of clickbait headlines by classifying it into one of two classes which are clickbait and non-clickbait. With making this detection available beforehand, online readers can be warned about potential clickbait links such that the readers can make their decision safely before clicking it. We aim to reproduce the work of Chakraborty et al. [3] by analyzing designing features that are a great measurement of clickbait headlines and using them to train machine learning models to detect clickbait headlines.

Chakraborty et al. [3] built an automatic clickbait detection using 14 features and testing with 3 classifiers. The features were categorized to sentence structure, word patterns, clickbait language, and N-gram features. These features were trained and tested with Support Vector Machines (SVM) with Radial Basis Function kernel, Decision Trees, and Random Forests. SVM had the best performance with an accuracy of 0.93, a precision of 0.95, and a recall of 0.9. Next, they investigated whether they can create a personalized clickbait blocker. They asked 12 regular newsreaders to label 200 clickbait headlines and whether they would click on the link. They built a clickbait blocker based on topical similarity, linguistic patterns, and the hybrid of the two. Clickbait blocker performed the best based on linguistic patterns with an accuracy of 0.81, a precision of 0.834, and a recall of 0.76.

In this study, we focused on reproducing specifically the clickbait detection by Chakraborty et al. [3]. Furthermore, some features from the original study were removed and other features were added. This paper looked at features from sentence structure, POS tagging, and sentiment analysis. We used XGBoost for feature selection to look at the importance of each of the features. For the classification task, SVM, Random Forest, and XGBoost were used. The classifiers were also cross-validated to prevent overfitting.

2 DATASET

The dataset from the experiment [3] conducted by Chakraborty et al. was used for this research. It consisted of 15999 clickbait headlines and 16001 non-clickbait headlines.

Non-clickbait headlines were retrieved from Wikinews, The New York Times, The Guardian, and The Hindu. Clickbait headlines were retrieved from BuzzFeed, Upworthy, ViralNova, Scoopwhoop, and ViralStories. The first letters of the words in the headlines were capitalized.

In the original experiment, 6 volunteers were recruited to label the headlines in the clickbait dataset whether they are clickbait or not clickbait, to reduce the risk of having non-clickbait headlines labeled as clickbait. The data was divided so that each headline would be labeled by at least 3 volunteers. In the end, 7623 headlines were labeled as clickbait. However, the manually labeled clickbait data was not provided by the researchers, and hence we have randomly selected 10000 clickbait and 10000 non-clickbait headlines from the original dataset.

The headlines were converted to lower case. The reason for the conversion was to avoid words being tagged as a proper noun during part-of-speech tagging for the features. The downside is that some proper nouns may not correctly be tagged, but since nouns and proper nouns are not used as one of the features, they should not be affected by this. The original experiment first retrieved the named entities, then converted the rest of the headline to lower case. However, this method of pre-processing would cause the same issues as not converting to lower case at all, since some words may be incorrectly identified as named entity and hence they would remain capitalized.

3 FEATURES

Extraction of all features except the sentiment analysis was done with spaCy version 2.2.3 with the model `en_core_web_lg`. Feature extraction was done on spaCy because it is very fast, and especially with a large dataset, computational complexity becomes a significant factor. The possible application of the clickbait classifier will most likely be applied to web-scale data, and hence faster computation may be favored. In [1], Al Omran and Treude claimed that out of the NLP libraries NLTK, spaCy, Stanford CoreNLP, and SyntaxNet, spacy and NLTK achieved the highest similarity in general part-of-speech (POS) tagging with manual annotation. Spacy also achieved the highest similarity with manual annotation for specific POS tagging. Since most of the features used in this study require POS tagging, spaCy was thought to give the least errors during feature extraction. For sentiment analysis, Stanford CoreNLP was used. More detail on sentiment analysis can be seen in 3. In the following, we present the used features for this study.

Sentence Structure

Number of words in a headline: It can be seen from Figure 1 that clickbait headlines are slightly longer than non-clickbait headlines. Non-clickbait articles aim to summarize the content, whereas clickbait articles need to make the headline interesting and attract as many readers as possible to gain access number. Hence, clickbait headlines often include longer phrases, such as "You Won't Believe ...", making the headline longer.

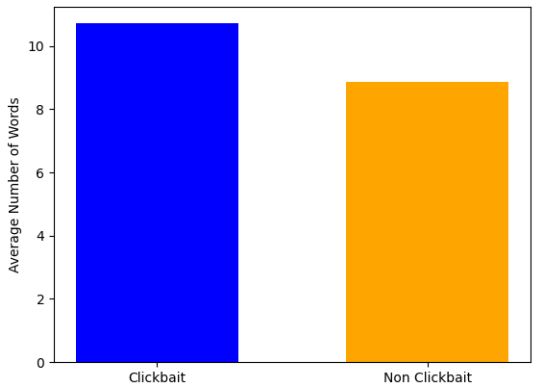


Figure 1: Difference in the average number of words between clickbait and non-clickbait headlines.

The average length of a word: Although clickbait headlines are longer, they tend to have smaller words. Figure 2 shows that although words in non-clickbait headlines had an average length of over 5, words in clickbait headlines have length below 4.5. This may be the case because non-clickbait headlines include more specific or technical words, which would give more context compared to clickbait headlines. Another reason may be that clickbait headlines often use informal abbreviated words such as "you'll" and "that'll", which causes the words to be shorter.

Words and Phrases

Percentage of stop words in a headline: Stop words, which do not carry a lot of context, tend to occur more often in clickbait headlines than in non-clickbait headlines. Figure 3 shows that clickbait headlines consist of over 40% of stop words, whereas non-clickbait headlines only have stop words usage below 25%. This may be the case because non-clickbait headlines aim to give the context of the article while being concise, and hence avoid usage of words that do not help readers understand the topic of the article.

Inclusion of a number in a headline: Numbers are much more often included in clickbait headlines than non-clickbait headlines. It can be seen from Figure 4 that numbers seem to be twice as more frequent in clickbait headlines than in non-clickbait headlines. The main use is for listing, such as "12

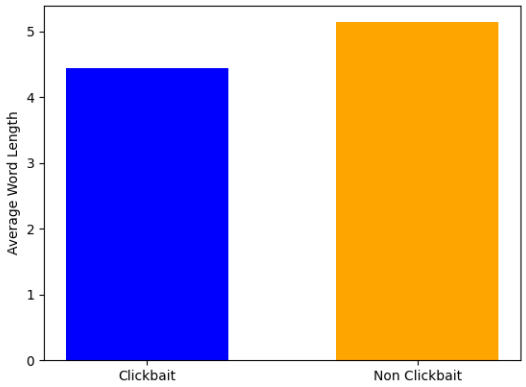


Figure 2: Difference in the average word length between clickbait and non-clickbait headlines.

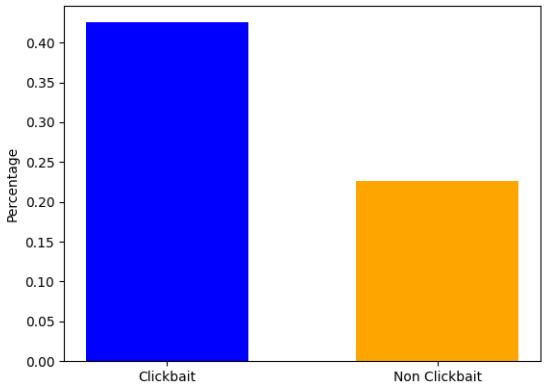


Figure 3: Difference in the average stop words usage between clickbait and non-clickbait headlines.

Mind-Blowing Ways To Eat Polenta". Non-clickbait headlines may include numbers, but it is rarer and it is used to add context.

Part-Of-Speech Tagging

Inclusion of a determiner in a headline: Determiners, for similar reasons to stop words, are used much more often in clickbait headlines than in non-clickbait headlines. Figure 4 shows the large difference in percentage between clickbait and non-clickbait headlines. Determiners do not add context and hence non-clickbait articles tend not to include them in the headline to make it more concise.

Inclusion of a pronoun in a headline: Figure 4 shows a major difference in the percentage of headline including pronouns between clickbait and non-clickbait articles. Although only 3% of non-clickbait headlines used pronouns, 55% of clickbait headlines included pronouns. Non-clickbait articles rarely used pronouns in the headlines because they seem to use proper nouns instead. Clickbait headlines try to attract readers to click the article by making the article more personal to the readers, such as "Are You Bart Or Lisa?".

Inclusion of a superlative adjective/adverb in a headline: Superlative words tend to be used more frequently in clickbait headlines to exaggerate the subject. Non-clickbait articles will less likely include superlative words since they can be quite subjective. It can be seen from Figure 4 that superlative words are not included often in general, but it is much more likely in a clickbait headline than a non-clickbait headline.

Inclusion of a comparative adjective/adverb in a headline: Similarly to superlative words, comparative words may be used more frequently in clickbait headlines for exageration. Figure 4 shows that although there is no major difference between the two classes, clickbait headlines had a slightly larger probability of including a comparative word.

Inclusion of a superlative and/or comparative adjective/adverb in a headline: Since both superlative and comparative words do not occur frequently and have a similar purpose of comparing, we decided to create a feature in which we look if headlines include either a superlative word or a comparative word. Figure 4 shows that clickbait headlines are much more likely to include these words. This feature may give duplicated information from the two features mentioned previously. The idea was that feature selection would allow keeping the more useful feature if they are useful at all.

Sentiment Analysis

Sentiment score: Based on the research by Chakraborty et al.[3], it is found that headlines that are considered clickbait are more likely to use words having Very Positive sentiment value such as "Inspiring", "Awesome", and "Breathtaking". While these words are almost non existent in headlines that are not clickbait. The use of these words in headlines can stimulate the impulse of the reader to click the follow up link hoping for sensational information. The first research about handling clickbait detection was done by Potthast et al. where it relied mainly on bag-of-words algorithm while also using sentiment analysis and some readability measures. They used sentiment polarity as a classifier feature for capturing the characteristics of a clickbait's teaser message or headline. Using sentiment analysis alone was insufficient to classify clickbait headlines but with the combination of other features, it gave some predictive power to the classifier.

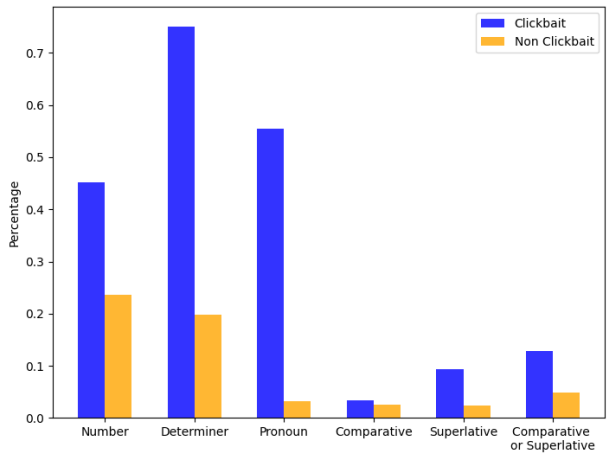


Figure 4: Percentage of including number, determiner, pronoun, comparative words, superlative words, and one of comparative and superlative words between clickbbait and non-clickbait headlines.

To do the sentiment analysis and extract the sentiment polarity or sentiment value for each headline, we use Standford CoreNLP library. It provides probabilistic parser for general text that has good accuracy. Most of sentiment analysis tools work by looking at words as a single entity. It does so by increasing the number of points if the sentence contain positive words and the decreasing the number of point for negative words. It then uses the sum of these points to give the sentiment value. The order of the words is not considered in this approach thus important information about the sentence might be lost. Stanford CoreNLP uses its deep learning model to build up a representation of sentences as a whole based on the sentence's structure thus it computes the sentiment value based on how the words compose meaning of the longer phrases [8]. This approach makes the model not as easily fooled by complex usage of words compared to the word count approach. The sentiment score range between 0 to 4 where 0 means "Very Negative" and 4 means "Very Positive". Looking at an example of sentence "This movie was actually neither that funny, nor super witty". Stanford CoreNLP model would learn that the word "funny" and "witty" are positive words but it will still give the sentences a negative sentiment as an overall result. Clickbait headlines use a lot of very positive words to attract readers attention while also often making use of negative words as the other part of the main content of the sentence. The standard word count approach will not be able to capture this pattern of clickbait headlines thus we use Standford CoreNLP. Figure 5 shows that there are very few headlines with 0 sentiment value for both clickbait and non-clickbait. While non-clickbait headlines dominates in having a sentiment value of 1 ("Negative"), clickbait headlines dominated in having a sentiment value of 3 ("Positive").

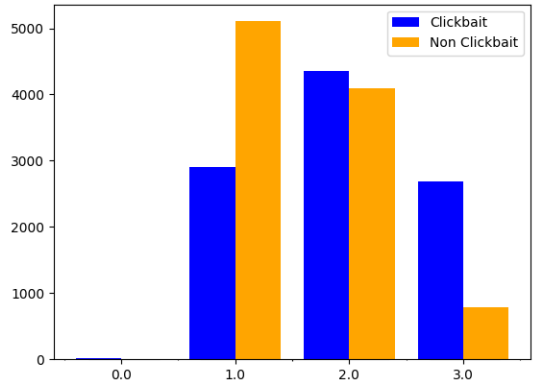


Figure 5: Count of sentences classified in different sentiment values between clickbait and non-clickbait headlines.

4 FEATURE SELECTION

Under the Occam's razor principle, if fewer features lead to the same result as using all, it is better to choose fewer features. Hence, we perform feature selection. We do this using XGBoost which allows generating of feature importance. The feature importance, in this study, is generated based on the number of times a feature is used for splitting trees such that the outcomes are more homogeneous (Ref. Section 5). Based on this implementation, Figure 6 was generated. From Figure 6 it can be seen that the features superlative, comparative and superlative or comparative do not contribute as much as other features. It was also observed that removal of these features lead to the same initial exploratory performance and hence the features were reduced by removing the aforementioned features.

The `XGBoost Release 1.1.0-SNAPSHOT` was used in this study. Specifically, `get_score(importance_type="weight")` was used to generate the importance of the features.

5 CLASSIFICATION ALGORITHMS

Detecting clickbait is a binary classification task. A headline can either be clickbait or not. The classifier's job is to project all the headlines into either the clickbait class or the non-clickbait class. It would do this by training on the features mentioned earlier. The `scikit-learn (V 0.22.2)` library was used in this study for classification.

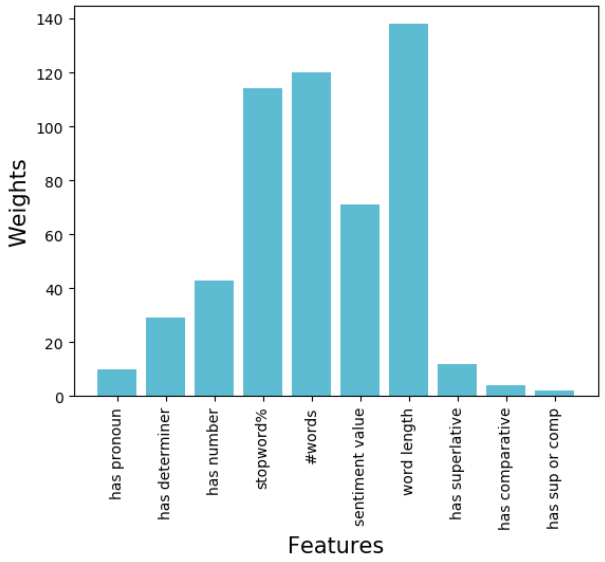


Figure 6: Feature importance generated by XGBoost. The x-axis depict the features and the y-axis depict the weights of each features.

Support Vector Machine

Much of the dimensions can be reduced with the help of feature selection, yet the data contains high dimensions. Consequently, this can lead to lower accuracies in classification and so Support Vector Machine (SVM) is used as a classifier [5]. A Radial Basis Function Kernel (RBF) is used since the features may not be linearly separable [1].

The basis of SVMs is to find a hyperplane that divides the classes the best. This hyperplane maximizes the distance between the hyper-plane and points in the training set [6]. SVMs also use a penalty parameter to state how much misclassification should be avoided.

`scikit-learn.svm.SVC` was used to implement the SVM classifier. The default value of the hyper-parameters was used.

Random Forest

With large datasets, there are higher chances of encountering noise. Hence, the Random Forest (RF) is used since it can control error internally [2].

RF contains a lot of decision trees with each tree predicting the class the observation belongs too. RF chooses the class based on the number of times a certain class is chosen for the observation. It is important that these trees are uncorrelated in order to make accurate predictions (being uncorrelated avoids internal error). RF ensures this uncorrelation by the use of bagging and feature randomness. Bagging is the random selection of observations from the dataset with replacement. Feature randomness involves a tree only picking from a random subset of features. This allows for variation between the decision trees which then leads to uncorrelation.

`scikit-learn.ensemble.RandomForestClassifier` was used to implement the RF classifier. The default value of the hyper-parameters was used. Although the version of `scikit-learn` used in this study has the default value of `10` trees, there were "`FutureWarning`" stating that the default would change to `100` trees and hence in the implementation `100` trees were used.

XGBoost

One of the most popular algorithms out there is XGBoost (Extreme Gradient Boosting). It has led to many state-of-the-art results in many fields including web text classification [4]. It is also considered to be computationally fast (due to parallelization) and hence, it was chosen for this study. The basis of this algorithm lies in boosting. Boosting involves taking weak learning models and transforming them into strong learning models. It iteratively modifies the weights of the weak learning models to improve its accuracy. XGBoost also has a stop criterion for tree splitting (splitting at nodes such that the outcomes are homogeneous) and hence can reduce its computation time again. This is unlike RF where one does not need to specify the maximum depth of the trees that are traversed. XGBoost is also able to learn feature relationships that are not explicit and it is also able to find optimal split points in the dataset.

`XGBClassifier` from the `xgboost` library was used to implement the XGBoost classifier. The default value of the hyper-parameters for the XGBoost of `scikit-learn API` was used.

6 EXPERIMENTS

Evaluation

Cross-Validation. For a classifier to perform well and be able to generalize, it needs to have a low bias and low variance. Bias is the difference between the average predicted headlines and the clickbait headlines that has to be predicted. Having a high bias would lead to oversimplification of the model and hence would lead to errors in classification of the data-sets. Variance is how much a headline varies, it provides the spread of the data. Having a high variance would lead to overfitting of the data and hence terrible generalization. Hence, the classifiers were cross validated.

The classifiers were cross validated with 10 folds. Cross validating helped reduce bias since a lot of the data was used

[1] http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex8/ex8.html

for fitting and also to reduce variance since it was validated on most of it too.

Accuracy, Precision, Recall, F1-Score. To evaluate the classifiers, it is important to not only look at the accuracy (Equation 1) which is the ratio of correctly classified clickbait headlines and total number of headlines. Solely looking at the accuracy would be appropriate only when there are the same number of false positives and false negatives. Hence, the F1 score (Equation 4) is determined which is the weighted average of precision (Equation 2) and recall (Equation 3). Precision is defined to be the ratio between true positives and the total number of clickbait headlines. Recall is defined as the ratio between true positives and all the tweets clickbait class. The F1 score is considered to be good if the values are far above 0.50. Hence, the classifier will be evaluated by looking at the accuracy, precision, recall and therefore also the F1 score.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

Where:

- TP : True Positives
- TN : True Negatives
- FP : False Positives
- FN : False Negatives

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{PN}} \tag{3}$$

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

Confusion Matrices, ROC, AUC. Confusion matrices can also be created which provides a summary of the value stated above. This can help in indicating the kind of errors that occur most often. These matrices can also be used to create the Receiving Operating Characteristics (ROC) curve. The ROC curve is plotted with True Positive Rate (TPR) against False Positive Rate (FPR). If the classifiers have a higher TPR then they are said to classify well. The Area Under this Curve (AUC) represents how well the clickbait headlines and non-clickbait headlines can be differentiated. If the AUC is 1 then the classification is said to be perfect, if the AUC is 0.5 then the classification is said to have happened by chance and the classifier is not considered to have a good performance.

7 RESULTS

Accuracy, Precision, Recall, F1-Score

Table 1 shows the accuracy, precision, recall and F1-score for each of the classifiers. The values are based on the data that was tested during cross validation and the values are averaged over the 10 folds. Looking at Table 1 it can be seen that the classifiers have fairly similar values.

Table 1: Accuracy, Precision, Recall and F1-Scores for each of the classifiers.

Classifier	Accuracy	Precision	Recall	F1-Score
SVM	0.868	0.886	0.846	0.865
RF	0.855	0.862	0.844	0.853
XGBoost	0.876	0.889	0.859	0.874

Confusion Matrices

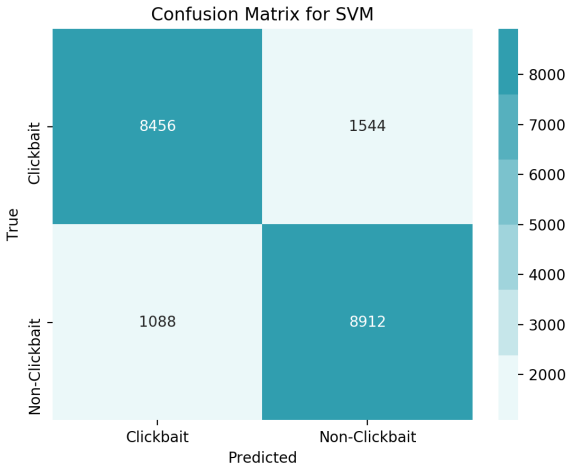
Figure 7 shows the confusion matrices for all the classifiers. The frequency of misclassifying clickbait as non-clickbait seems to be higher than misclassifying non-clickbait as clickbait for the all classifiers.

Receiver Operating Characteristic

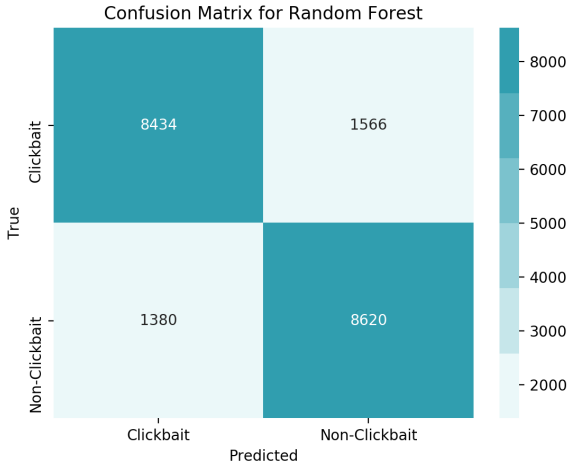
The Receiver Operating Characteristic (ROC) curves can be seen in Figure 8. The Area Under Curve (AUC) can also be seen in Figure 8 for each of the classifiers. The classifiers seem to have similar AUCs and seem to perform fairly well.

8 DISCUSSION

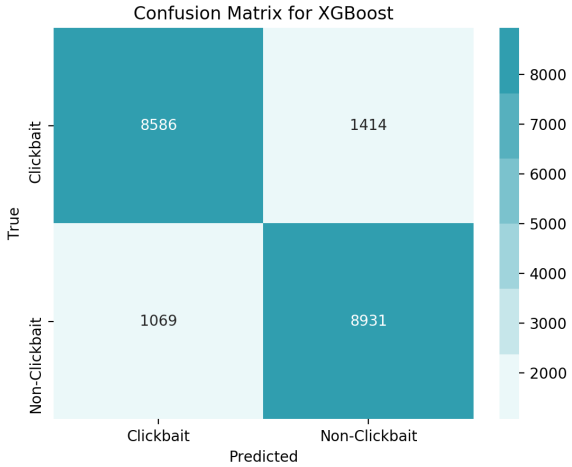
The purpose of this study was to determine which features are useful in predicting clickbait headlines. Based on the results, it seems that superlative and comparative words are unimportant features when it comes to classifying clickbaits as seen in Figure 6. It can also be seen that numerical features, such as the number of words or the word length, tend to contribute more than binary features, such as whether the



(a) SVM: TP=8456 TN=8912 FP=1088 FN=1544



(b) RF: TP=8434 TN=8620 FP=1380 FN=1566



(c) XGBoost: TP=8586 TN=8931 FP=1069 FN=1414

Figure 7: Confusion matrices of all the classifiers.

headlines have pronoun or whether the headlines have determiners present. This may indicate that using distribution of POS-tags as features might be more useful than simply checking the presence.

The classification results show that the classifiers are able to predict clickbait headlines rather well. The F1-scores of all of them lie between 0.85 to 0.87 which provide evidence of good classification. Also, from Figure 8 it can be seen that the AUCs lie between 0.85 to 0.87 which again provide evidence of good classification. Looking at the confusion matrices shown in Figure 7, it can be seen that the frequency of misclassifying clickbait as non-clickbait seems to be higher than misclassifying non-clickbait as clickbait for the all classifiers. This may be the case because some headlines labeled clickbait may not actually be clickbait headlines. As is explained in 2, the study by Chakraborty et al. [3] had volunteers manually label the clickbait dataset to check whether they were really clickbaits. However, the manually labeled data was not provided, and so it could be the case that the higher frequency of misclassifying clickbait as non-clickbait is a consequence of mis-labelling of the data instead of the actual classification technique.

Comparing the results of this study to Chakraborty et al. [3], it can be seen that the performance of the classifiers in [3] is better than the performance of the classifier in this study. Namely, the F1-scores of the SVM and RF classifiers in [3] are higher than the ones in this study. One reason why this may be so is because, in this study, the hyper-parameters were not tuned which might not be the case in [3], albeit there were no mentions of any tuning. As mentioned earlier, they also used manual labelling of data which was not provided, hence the difference in classification performance of the same classifiers. Yet another, but important reason for this difference lies in the features used to train the classifiers. They used a total of 14 features to classify, whereas, this study used 7 features to classify. The main focus of this study in terms of features lie on POS-tagging, sentence structure as well as sentence sentiment, whereas in [3], they also look at N-grams. They show that using N-grams leads to good performance of the classifiers and hence, this may also be another reason for the difference in performance.

9 RESOURCE REPOSITORY

The code and dataset used for this study can be found here: https://github.com/ShoCremers/Applied-NLP-2020

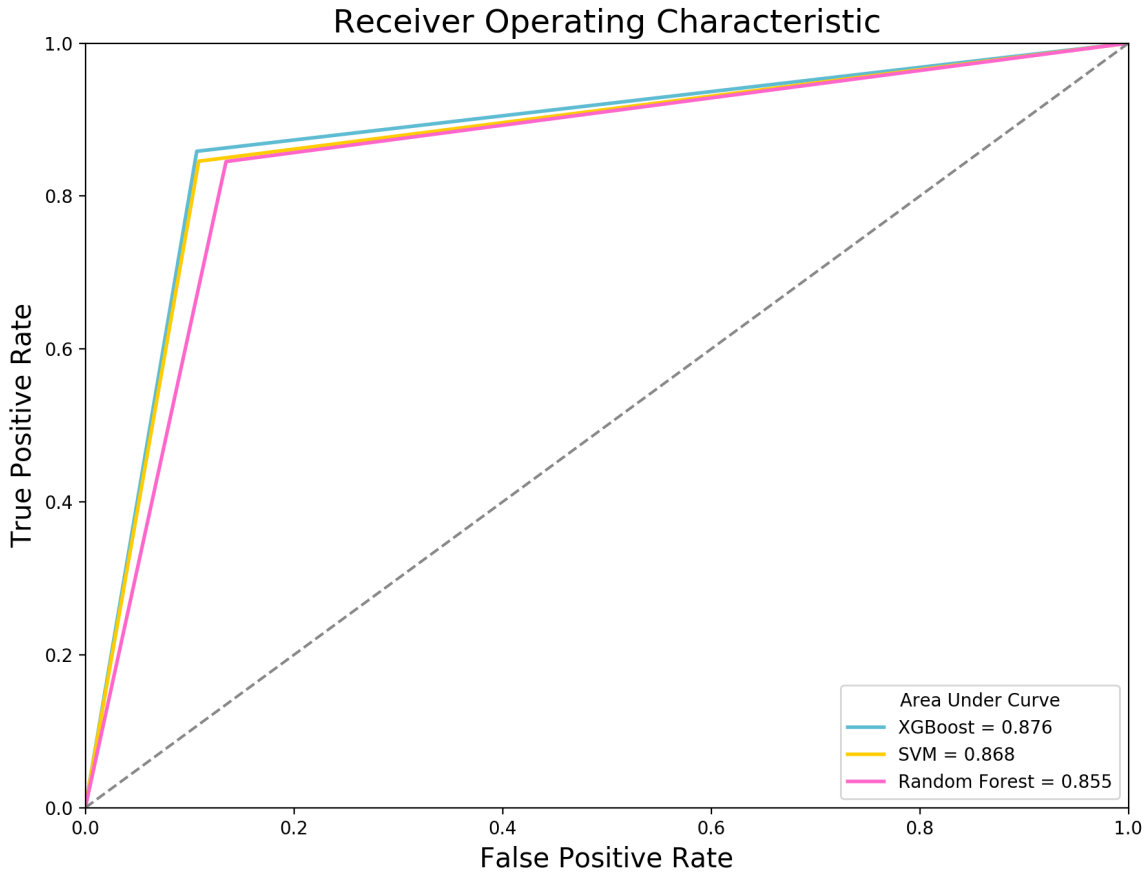


Figure 8: Receiver Operating Characteristic for each of the classifier; SVM (blue), RF (yellow), XGBoost (pink). The x-axis show the false positive rate and the y-axis show the true positive rate. The grey dashed line depicts classification by chance.

REFERENCES

[1] F. N. A. Al Omran and C. Treude. 2017. Choosing an NLP Library for Analyzing Software Documentation: A Systematic Literature Review and a Series of Experiments. In *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*. 187–197. https://doi.org/10.1109/MSR.2017.42

[2] Leo Breiman. 2001. *Machine Learning* 45, 1 (2001), 5–32. https://doi.org/10.1023/a:1010933404324

[3] Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop Clickbait: Detecting and preventing clickbaits in online news media. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE. https://doi.org/10.1109/asonam.2016.7752207

[4] Tianqi Chen and Carlos Guestrin. 2016. XGBoost. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*. ACM Press. https://doi.org/10.1145/2939672.2939785

[5] Corinna Cortes and Vladimir Vapnik. 1995. *Machine Learning* 20, 3 (1995), 273–297. https://doi.org/10.1023/a:1022627411411

[6] Jurij Leskovec, Anand Rajaraman, and Jeffrey David Ullman. 2015. *Mining of massive datasets*. Cambridge University Press.

[7] Martin Potthast, Sebastian Köpsel, Benno Stein, and Matthias Hagen. 2016. Clickbait Detection. In *ECIR*.

[8] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, 1631–1642. https://www.aclweb.org/anthology/D13-1170