NEWS TEXT CLASSIFICATION

Project Report

Submitted by:

UMANG KAMDAR - AU1841069

YESHA AJUDIA - AU1841078

KARTAVI BAXI - AU1841079

JAINESH PATEL - AU1841101

HETVI PAREKH - AU1900003

at

Ahmedabad University

School of Computer Studies (SCS)

Ahmedabad, Gujarat

Table of Contents

List of Figures

1. INTRODUCTION

1.1. Problem Definition

Increase in the digitization and the number of phone and similar users has led to various portals containing information from news, blogs, articles and ebooks. For a frequent user of such digital platforms, it would be really helpful to get desirable information without the need to scroll to large lists and tons of sources. Multiple subject labels are often used to report news articles. For example, in the context of sport, business, and world news a report on transfer ownership of the football club might be marked. Users prefer to read articles/news, based on their areas of interest. And hence applications and websites use various ways to sort the articles based on different categories which makes it comfortable for users to find them [10]. People can accurately recognize and give various noteworthy labels for an article, but can a machine learning framework deliver comparable findings?

The machine learning model for automated category-wise text classification could be used to identify topics of untracked news and make individual suggestions based on the user's prior interests [14]. Text classification offers a good framework for getting similar content with textual data processing without lacking its uniqueness [15].

1.2. Project Overview and Specifications

The project is focused on the development of a classification system, focused on the classification of news articles into a few predefined categories. This project uses a hybrid model—machine learning-based classification in combination with rule-based classification. It is built upon the concepts of Natural Language Processing. The developed system tries to assimilate a rule-based approach also, in order to eliminate the flaws, if any, in the model trained by only the machine learning approach. The project also focuses on optimising the classification-prediction process by developing a system capable of classifying the input more than once by modifying the input through different modules of the system, comparing the different prediction outputs and then, providing the final prediction, based on the coded conditions.

The developed system can be a feature for various digital platforms; news publications and repositories. Since this system can classify the articles, it can be used to provide suggestions to the user based on the intent and interests of the users. It can even be used to classify the dump of articles into repositories and can also help in curating categorised datasets. Along with various other features and writing tools, this can also be a utility package for the journalists who can check the domain tone (political, entertainment) of their written article by testing it through this system. Thus, it is likely for the project and the developed system to have multiple uses.

2. LITERATURE SURVEY

2.1. Existing Systems

Classification is a difficult activity as it requires preprocessing steps to convert the textual data into structured form from the un-structured form. Text classification process involves following main steps for classification of news articles: data collection, pre-processing, feature selection, classification techniques application, and evaluating performance measures [9]. Text preprocessing is a vital stage in text classification particularly and text mining generally. It can be done by text documents collection, tokenization, stop-words removal, and stemming [4]. The frequency distribution of every unigram and bi-gram gives the idea about the frequent words appearing in all categories. N-grams are a viable alternative to words as indexing terms in information retrieval. N-grams provide higher accuracy than a strawman system using raw words as indexing terms [5]. It is plausible to use both—unigrams and bigrams, since there is risk of underfitting when only unigrams are used. Additionally, if N-grams—other than unigrams and bigrams are used, then there is risk of overfitting and also, the program might become computationally heavy. Data splitting is dividing the dataset into two subsets – one subset is used for training while the other subset is left out and the performance of the final model is evaluated on it. Data splitting is performed to train the model on a specific data set and then test it on unseen data to get the best understanding of accuracy [6]. All the classification algorithms have their own pros and cons and a good performance on classification demands the right choice of classifier for the right problem [2]. Bag of Words just creates a set of vectors containing the count of word occurrences in the document, while the TF-IDF model contains information on the more important words and the less important ones as well. Bag of Words vectors are easy to interpret. However, TF-IDF usually performs better in machine learning models [11].

Machine learning techniques that have been actively explored for text classification include Naive Bayes classifier, K-nearest neighbor classifiers, support vector machine, neural networks [1].

Naive Bayes is a simple probabilistic classifier which works on the assumption of conditional independence between the features of a text document [2], which is the so-called "Naive Bayes assumption" [8]. Naive Bayes has been successfully applied to document classification in many research efforts [8]. When compared to the state-of-the-art algorithms for decision tree induction, instance-based learning, and rule induction on standard benchmark datasets, Naive Bayes is found to be sometimes superior to the other learning schemes, even on datasets with substantial feature dependencies [3]. It is based on the Bayes rule which says

$$P(c|d) = (P(d|c) \, * \, P(c)) \, / \, P(d)$$

where *P(c|d),* is the probability of document *d* to belong to class *c*, called the posterior probability, *P(d|c)* is the likelihood and *P(c)* is the prior. Naive Bayes classifies the document to the class which maximizes the posterior probability [2]. There are mainly two models for Naive Bayes classification, they are the Multivariate Bernoulli Model and the Multinomial Model [12]. The Multivariate model has binary representation of features while the later represents the features with term frequency [2]. It was observed in [13] that Multinomial Naive Bayes performs better than Multivariate Bernoulli Naive Bayes. Multinomial Naive Bayes event model is more suitable when the dataset is large when compared to the Multivariate Bernoulli Naive Bayes Model [7].

2.2. Proposed System

The system proposed, includes some more features than the existing work. As per the knowledge of the authors, so far no literature has combined the classification and summarization tasks. The proposed system not only performs classification on test data after cleaning, but it also takes some input from the user, classifies the given article into either one of the predefined classes or a category named 'Others', i.e., the article does not belong to any of the predefined categories. The article being classified into 'Others' category is based on a threshold value which is decided after several observations.

Further, to improve the classification results, the given input is summarized to one-third of the length of the original article using a text summarization technique, and a class for the summary is predicted, same as the prediction done previously. This results into two classes, either the same or different. The final prediction result is decided based on some predefined rules: If both prediction results into the same category, the final prediction is considered as any of them, if any one of them results into 'Others' category, the final prediction will be 'Others' category, and if both results into any of the predefined categories, but both are not same, then it is considered a case of multi-label classification and the article is classified into both the categories.

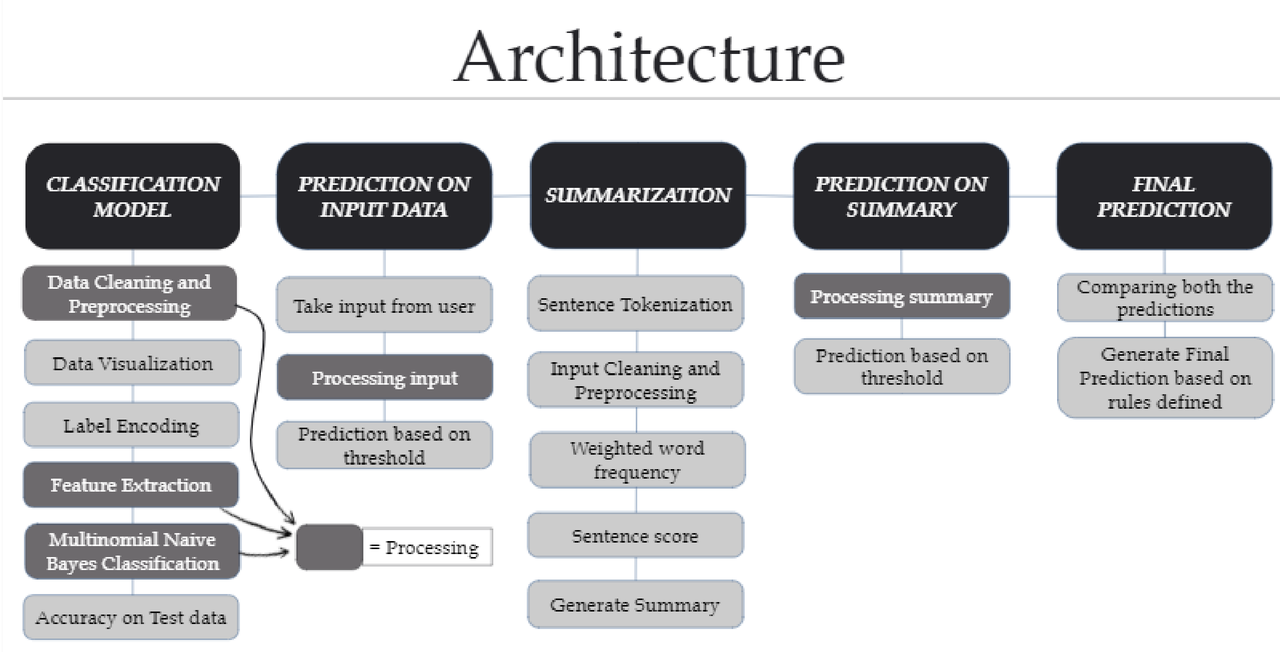

Figure 1: Proposed System

3. SYSTEM ANALYSIS & DESIGN

3.1. Algorithm and Pseudo Code

Following libraries have been imported for implementing the project:

```
import numpy as np    # importing numpy lib; used for working with arrays
import pandas as pd   # importing pandas lib; used for analysing data & manupulating it
import os             # used for interacting with the OS
import itertools      # used for doing all kinds of complex iterations
import re             # regular expression: provide matching operations
%matplotlib inline
import matplotlib.pyplot as plt     # used for plotting the graphs
from nltk.corpus import stopwords   # list of unwanted words of english
from nltk.corpus import wordnet     # a database of English Nouns, Adjectives, Adverbs and Verbs
from nltk.corpus import words       # includes list of english words
from nltk.stem import WordNetLemmatizer       # used for morphological analysis of the words; {good: better, best}
from nltk.tokenize import word_tokenize         # used to divide text into smaller part called tokens
```

Figure 2: Libraries Used

The proposed system has the following steps for article classification:

3.1.1. Data Importing

Here, two different datasets are used to get the balance between the various news categories. One dataset is taken from Harvard Dataverse which is publicly available. It includes features like the publish date, title, subtitle and text. This dataset contains the news from various news sources like ABC News, CNN news, The Huffington Post, BBC News, DW News, TASS News, Al Jazeera News, China Daily and RTE News. The articles which fall under the categories that did not have enough number of samples were removed from the dataset and the categories of the remaining samples were defined.

Now the category distribution was uneven, which might have led to misclassification. So, we added some samples from the publicly available BBC News dataset which had articles different from the one included in the Harvard dataset. Hence, now we have a total of 2509 samples in the dataset, which fall under five categories namely, politics, sports, entertainment, business, and tech.

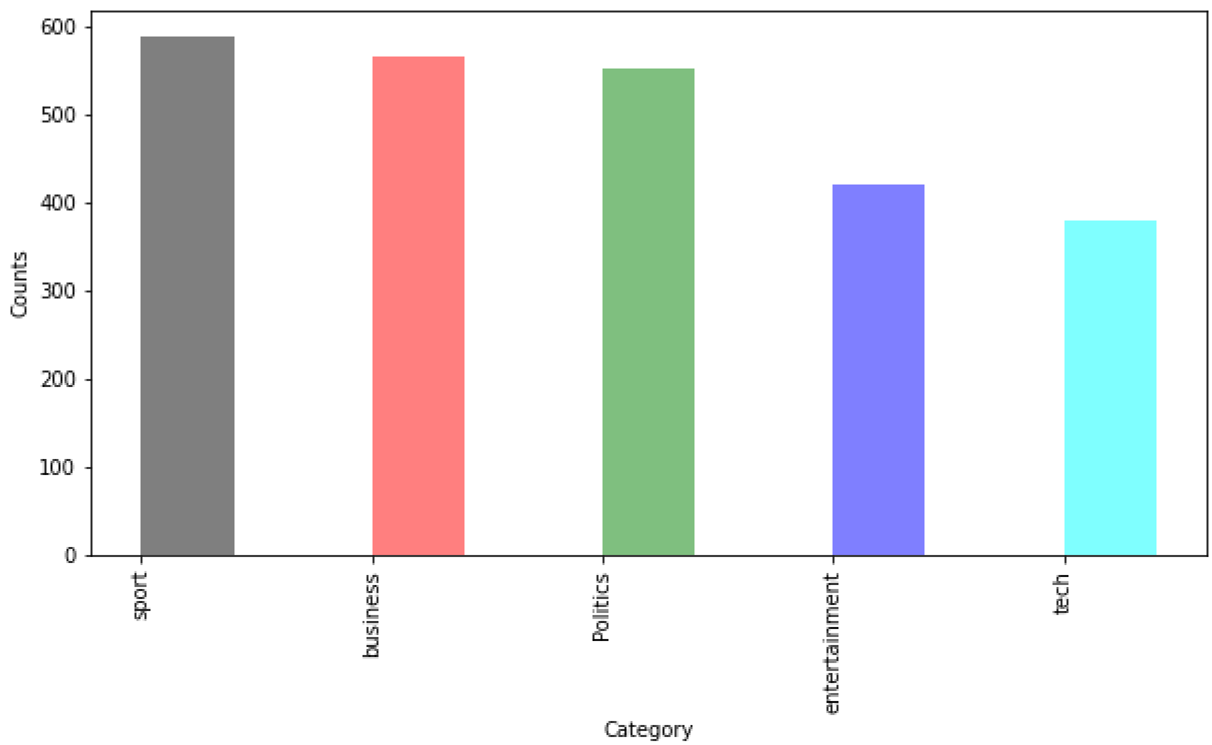

Figure 3: Data Balance Chart

3.1.2. Data Cleaning, Preprocessing and Visualization

Data cleaning was the very next step after importing, where we defined a function that removes the unwanted spaces from the news text, change the cases from uppercase to lowercase, replaces all the characters other than A-Z/a-z with space, replace some short form words to its full-forms, etc. All the cleaned text after this step was stored in a new column named ‘Cleaned’. There were some unwanted columns like article_id, publish_date, article_source_link, which were not required for classification. Hence, such columns were removed.

The next step was stop words removal and lemmatization, for which a separate function is defined. Hence, the stopwords list of the ‘nltk library’ was used. Stopwords are the words that occur frequently but do not add much value to analyse the text. The cleaned data was then tokenized and lemmatized. Lemmatizing means converting the word to its root word. It is useful to reduce the number of words and to analyze the text more clearly. A separate column named ‘Cleaned_with_lemma’ was created to store the lemmatized text.

After data processing, we plotted the frequency distribution of words in each category for unigram and bigrams, to visualize which words fall under each category and how the words are distributed.

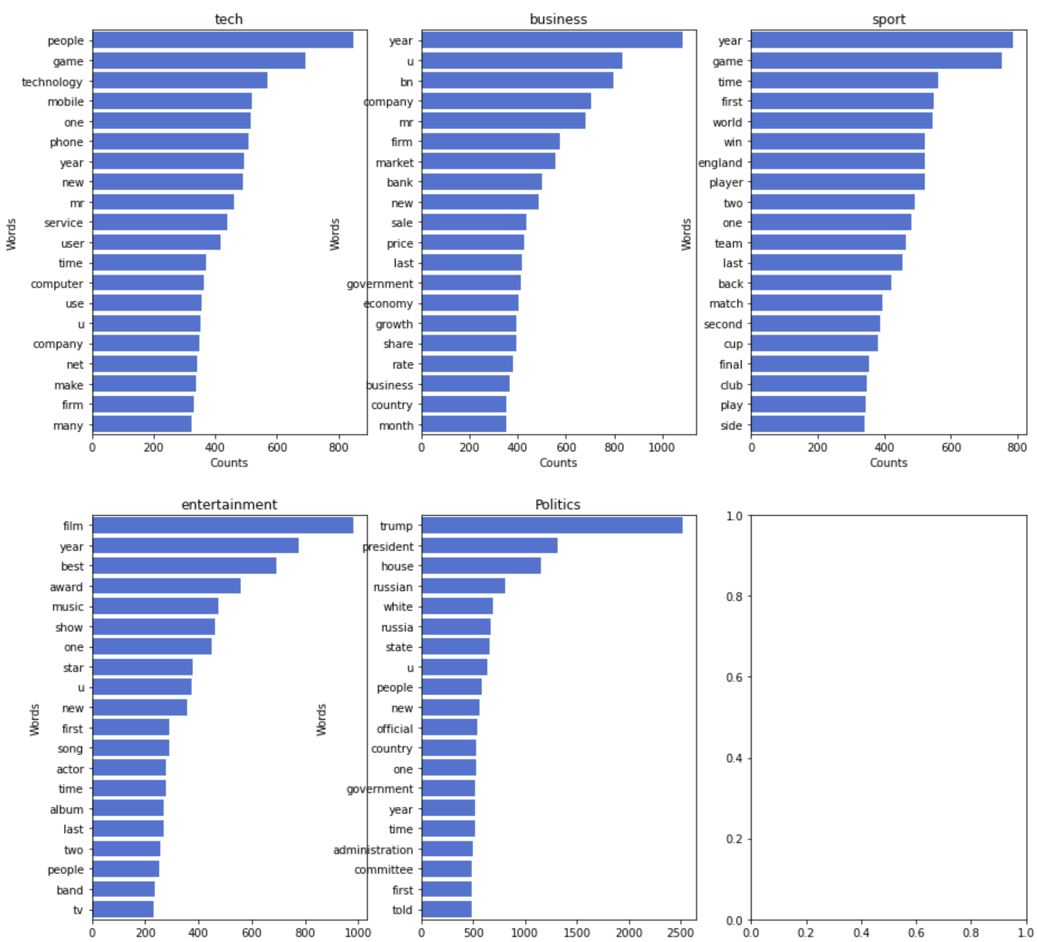

Figure 4: Unigram Frequency Distribution

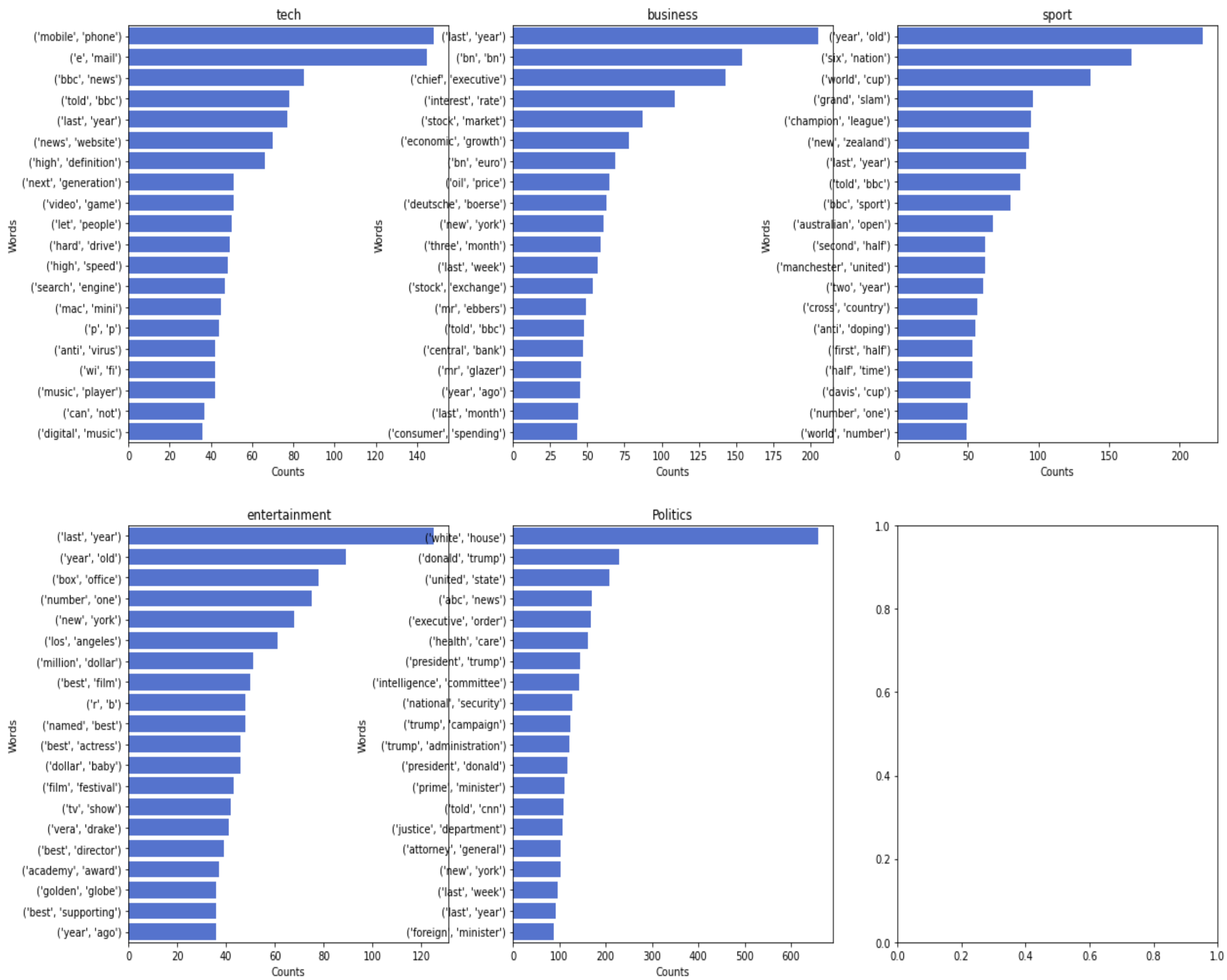

Figure 5: Bigram Frequency Distribution

3.1.3. Label Encoding and Data Splitting

Label Encoding is used to convert labels to numerical form so that it becomes easily readable by machine learning algorithms. We encode target labels with value between 0 and n_classes-1. The encoding in our case is as follows:

Politics	0
business	1
entertainment	2
sport	3
tech	4

Next, we have defined the input features and output labels, by splitting the data into training and testing to fit our model. 80% of the data is used for training and the rest 20% is used for testing, by randomly selecting from the dataset.

3.1.4. Feature Extraction

A TF-IDF vectorizer is used to transform the text into feature vectors. Each unique word becomes a feature, and a feature vector is defined for each example. In each vector the number defines the TF-IDF score of that feature. These feature vectors are used as input for machine learning algorithms. For our application, the TF-IDF vectorizer uses unigrams and bigrams to generate the feature vectors for both training dataset and testing dataset.

3.1.5. Fit Classification Model

Once we have the feature vectors, the next step is to define a classifier. We have used the Multinomial Naive Bayes classifier to train our model. Feature vectors and labels are given as input to the classifier to learn from the dataset and fit the model well. Predictions are made for the test dataset and the accuracy score of the model is computed for the test data. Confusion matrix and classification report are generated to measure the performance metrics.

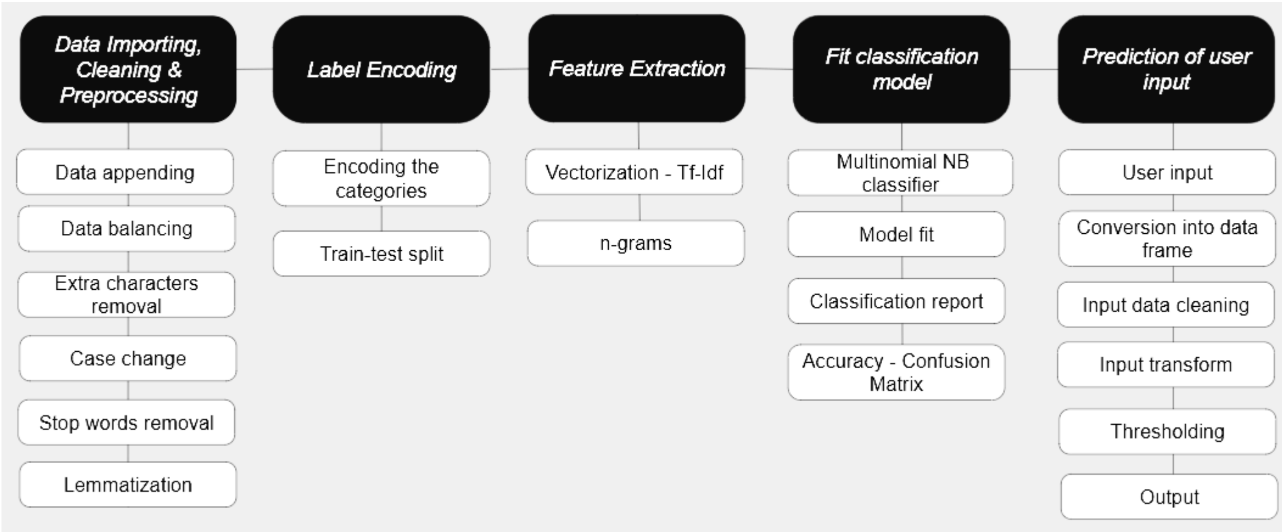

Figure 6: Classification Modelling

3.1.6. User Input Prediction

A test input from the user is taken, and the classification model developed is used to predict the category of the given input.

There might be a case where the input news text from the user doesn't belong to any of the five predefined categories. For this, the algorithm has been modified such that in such a case, the system gives 'Others' as prediction output. This is done by comparing the highest probability with a threshold value, which decides whether the input belongs to any of the predefined categories or to the 'Others' category. The threshold value is selected after testing many news articles with categories other than the five predefined categories (like 'Health', 'Fashion' etc.) from various online sources. The final threshold value has been fixed as 0.35. To have a more accurate classification, 'Text Summarization' technique has also been used for the process of classification.

For summarization, the text is tokenized into sentences to find the total number of sentences in the text. The necessary data cleaning steps are performed and the weighted frequency of words is found by dividing the frequency of each word with the maximum frequency. Then the sentence score of each sentence is calculated by summing up weighted word frequencies of individual sentences. The summary is then generated based on one-third of the total number of sentences with the highest sentence scores. The category of the summarized text is also predicted using the same method as above, by comparing the highest probability with a threshold value.

3.2. Testing Process

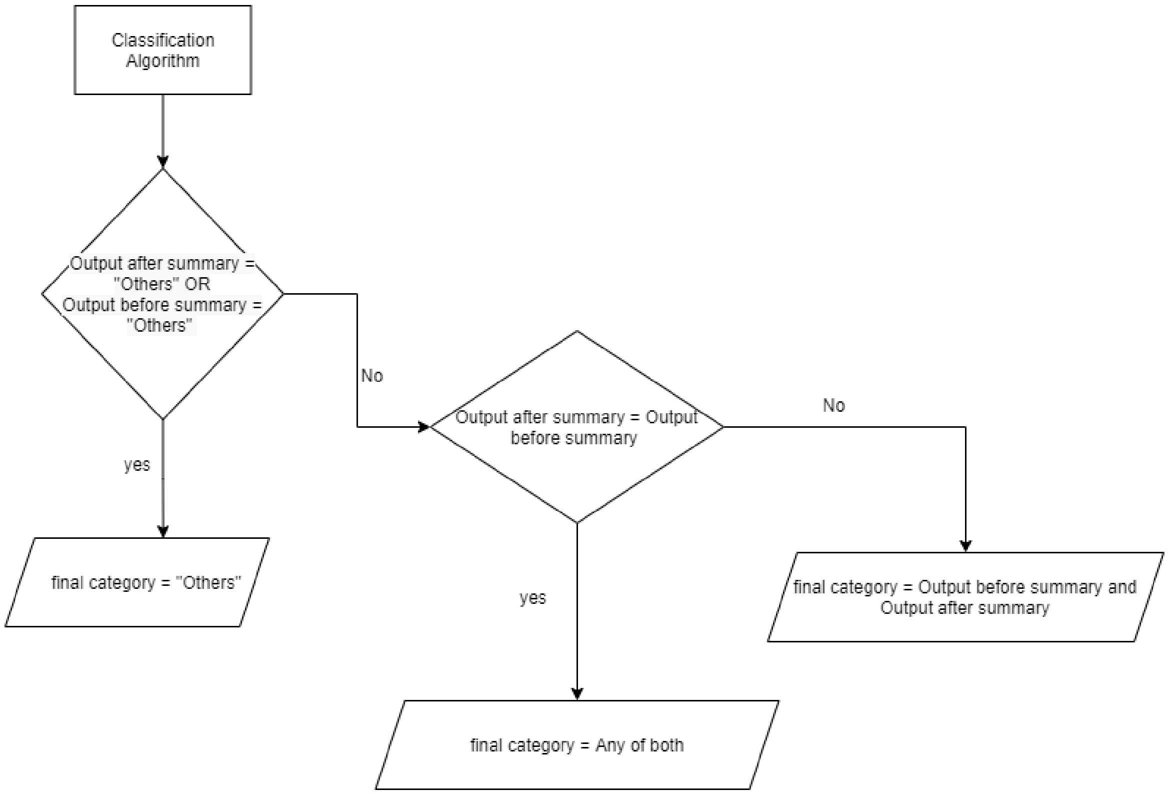

Figure 7: Rule Defining Flowchart

Now, the process of classification-prediction gets divided into two parts—prediction before summarization and prediction after summarization. For predicting final class, a rule-based strategy has been used, which involves both the predictions made above. If both the results are the same i.e the category prediction for both normal (before summarization) and summarized text is same, then the final prediction remains the same. If any one of the predictions is 'Others', then the final prediction of the input text will be 'Others'. If both results belong to any of the predefined categories but are not the same then we consider this as a case of multi-label classification, where the article is classified into two categories.

4. RESULTS & OUTPUTS

	precision	recall	f1-score	support
0	0.85	1.00	0.92	111
1	0.93	0.97	0.95	113
2	0.99	0.83	0.90	84
3	0.96	0.99	0.97	118
4	1.00	0.79	0.88	76
accuracy			0.93	502
macro avg	0.94	0.92	0.93	502
weighted avg	0.94	0.93	0.93	502

```
Accuracy of model on testing data is 0.9322709163346613
F1 Score of model on testing data is 0.9260630143819725
Log loss of model on testing data is 0.37458212185733425
```

Figure 8: Accuracy Report

After training the model with train data and testing the model with the test data, accuracy of around 93% was achieved by employing Confusion Matrix. The perfect precision score was achieved for category 4 (Tech) and a perfect recall score was achieved for category 0 (Politics). However, the highest f1-score was achieved for category 3 (Sport). One possible reason for better accuracy for the sport category can be the highest number of articles belonging to the sport category were present in the dataset.

After the user puts in an article in the input box allotted for input data, the same processes reoccur and the input article is classified into a category out of the five categories or into the 'Others' category. As this classification happens in two parts: classification before summary and classification after summary, the following output cases are possible:

```
Enter the test input: NEW YORK (AP) — Carrie Underwood brought the Academy of Country Music Awards to
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.05998366 0.05494141 0.67513902 0.16115775 0.04877816]]
-----CODE-----
Category of article:
entertainment

Highest Probability:  0.6751390192038647
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.08050692 0.08530141 0.52991854 0.2295097  0.07476342]]
-----CODE-----
Category of article:
entertainment

Highest Probability:  0.5299185414637095
-----CODE-----
Final prediction after comparison: entertainment
```

Figure 9: Output Case (1)

```
Enter the test input: Refillable skincare packaging seems like a no-brainer, but the practical challeng
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.25034135 0.27679227 0.09818297 0.19091478 0.18376865]]
-----CODE-----
Category of article:
Others

Highest Probability:  0.2767922650886336
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.25527156 0.22830044 0.13183794 0.2105136  0.17407646]]
-----CODE-----
Category of article:
Others

Highest Probability:  0.25527156145820245
-----CODE-----
Final prediction after comparison: Others
```

Figure 10: Output Case (2)

When the above output is received *(output case (1) & (2))*, i.e. in both the parts, the article is classified into the same category—either into a predefined category or into the 'Others' category. In this scenario, the final prediction is the category classified in both the parts as the result is the same for both the parts and there is no ambiguity in the classification.

```
Enter the test input: 4. The beginning of 2021 brought with it some good news about cinema halls reopen
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.14322144 0.15196904 0.40074605 0.15023231 0.15383117]]
-----CODE-----
Category of article:
entertainment

Highest Probability:  0.4007460525646401
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.19413525 0.18652857 0.29584977 0.18046592 0.14302048]]
-----CODE-----
Category of article:
Others

Highest Probability:  0.2958497747189966
-----CODE-----

Final prediction after comparison: Others
```

Figure 11: Output Case (3)

```
Enter the test input:  A rhythmic thumping sound echoed through the restaurant-lined street in north-we
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.26838845 0.28758313 0.12184469 0.20873597 0.11344776]]
-----CODE-----
Category of article:
Others

Highest Probability:  0.2875831309246079
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.33560889 0.3854942  0.06626722 0.14462935 0.06800034]]
-----CODE-----
Category of article:
business

Highest Probability:  0.38549420020083947
-----CODE-----

Final prediction after comparison: Others
```

Figure 12: Output Case (4)

When the above output is received *(output case (3) & (4))*, i.e. when the article is classified into the ‘Others’ category either in the former or the latter part, and in the other part, it is classified into one of the predefined categories, then the final prediction is given as ‘Others’. It is to eliminate the ambiguity and put it under a broader umbrella of the ‘Others’ category.

```
Enter the test input: Industry chiefs and strategic experts on Monday were unanimous in their view that
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.36082114 0.37069704 0.05421531 0.14815242 0.0661141 ]]
-----CODE-----
Category of article:
business

Highest Probability:  0.370697037344432
-----CODE-----
Data Cleaned
Stop words removed and Lemmatized
Probability distribution among categories:  [[0.38828854 0.19148286 0.13653587 0.17095663 0.11273609]]
-----CODE-----
Category of article:
Politics

Highest Probability:  0.38828854157854575
-----CODE-----

Final prediction after comparison: business-Politics
```

Figure 13: Output Case (5)

When the above output is received *(output case (5))*, i.e. when the article is classified into two different predefined categories in the former and the latter parts, the final prediction is given as a multi-label classification—category predicted in the former part and the category predicted in the latter part. In such a scenario, it is very likely that in either or both the parts, there might not be a significant difference between the two predefined categories and hence, two different predictions for the predefined categories in both parts. So, multi-label classification justifies both the categories with significant probabilities and classifies the article on a more granular level.

5. CONCLUSION & RECOMMENDATIONS

Our findings also align with the existing literature. Multinomial Naive Bayes seems the most appropriate method for the classification of news articles. Moreover, TF-IDF is also better than the Bag-of-Words model for feature extraction in this case because it shows the significance of both—more important and less important words.

Moreover, as seen, the prediction might differ after the article is reduced into the summary; leaving only the core elements. Hence, taking into consideration the prediction before and after the summary, comparing them and then providing the final prediction as the output with the help of defined rules is a better way in order to optimise the classification programme. Thus, this is a hybrid model—machine learning-based and rule-based.

Since the Multinomial Naive Bayes classifier is implemented, it is based on the assumption of conditional independence—features are independent of each other given their class. Hence, for the cases where this assumption is falsified, the accuracy might affect. This leaves scope for a classification method that works regardless of this assumption. With a larger dataset, more categories can be added for training the model, so as to increase the classification scope of the project. Moreover, multi-label classification is implemented for only one case and not for all the cases as a scope of this project. Multi-label classification can be a better representation of article classification-prediction and might classify in a more granular manner. This project classifies the article into the 'Others' category but is unable to recognise whether the article input by the user is a news article or a simple text corpus and classifies everything that is given as input. Thus, it can be programmed to identify and not accept input other than news articles. The classification into the 'Others' category for both the parts, as seen, is done on the basis of a threshold value that has been set on the basis of observation and testing a number of articles. However, it can be made more dynamic with a larger dataset and with more conditioned rules. Apart from these recommendations, which can help to optimize the system, more features, such as sentiment analysis, can be added along with the classification model with a front-end for a complete utility system for any particular user segment.

6. REFERENCES

[1] Patra, Anuradha, and Divakar Singh. "A survey report on text classification with different term weighing methods and comparison between classification algorithms." International Journal of Computer Applications 75.7 (2013).

[2] Vijayan, Vikas K., K. R. Bindu, and Latha Parameswaran. "A comprehensive study of text classification algorithms." 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, 2017.

[3] Domingos, Pedro, and Michael Pazzani. "On the optimality of the simple Bayesian classifier under zero-one loss." Machine learning 29.2 (1997): 103-130.

[4] Kadhim, Ammar Ismael. ”An Evaluation of Preprocessing Techniques for Text Classification.” International Journal of Computer Science and Information Security 16.6 (2018).

[5] McNamee, P., Mayfield, J. Character N-Gram Tokenization for European Language Text Retrieval. Information Retrieval 7, 73–97 (2004).

[6] Reitermanova, Zuzana. ”Data splitting.” WDS. Vol. 10. 2010.

[7] ghila, G. ”A Survey of Naive Bayes Machine Learning approach in Text Document Classification.” arXiv preprint arXiv:1003.1795 (2010).

[8] McCallum, Andrew, and Kamal Nigam. ”A comparison of event models for naive bayes text classification.” AAAI-98 workshop on learning for text categorization. Vol. 752. No. 1. 1998.

[9] Kaur, Gurmeet, and Karan Bajaj. "News classification and its techniques: a review." IOSR Journal of Computer Engineering (IOSR-JCE) 18.1 (2016): 22-26.

[10] Krishnamoorthy, Arjun, et al. "News Article Classification with Clustering using Semi-Supervised Learning." 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, 2018.

[11] Text, Q. and Huilgol, P., 2021. BoW Model and TF-IDF For Creating Feature From Text. [online] Analytics Vidhya. Available at: ¡https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/¿ [Accessed 17 March 2021].

[12] Aggarwal, Charu C., and ChengXiang Zhai. "A survey of text classification algorithms." Mining text data. Springer, Boston, MA, 2012. 163-222.

[13] Singh, Gurinder, et al. "Comparison between multinomial and Bernoulli Naive Bayes for text classification." 2019 International Conference on Automation, Computational and Technology Management (ICACTM). IEEE, 2019.

[14] British Broadcasting Corporation (BBC). “Insight - BBC Datasets.” Insight Resources - BBC Datasets, BBC, 2006, mlg.ucd.ie/datasets/bbc.html.
[15] “Text Classification: The First Step Toward NLP Mastery.” Medium, 12 Sept. 2020, medium.com/data-from-the-trenches/text-classification-the-first-step-toward-nlp-mastery-f5f95d525d73.

7. APPENDIX: IMPORTANT TERMS

C

- Confusion Matrix: Confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

F

- F-measure: Scores of precision and recall are combined and balanced into the calculation of the F-Measure.

P

- Precision: Precision is calculated as the sum of true positives across all classes divided by the sum of true positives and false positives across all classes.

R

- Recall: Recall is calculated as the sum of true positives across all classes divided by the sum of true positives and false negatives across all classes.

S

- Support: Support is the number of actual occurrences of the class in the specified dataset.

*