

Abstract

Fake News has become one of the major problems in the existing society. Fake News has high potential to change opinions, facts and can be the most dangerous weapon in influencing society.

The proposed project uses NLP techniques for detecting the 'fake news', that is, misleading news stories which come from non-reputable sources. By building a model based on a K-Means clustering algorithm, the fake news can be detected. The data science community has responded by taking actions against the problem. It is impossible to determine if the news is real or fake accurately. So, the proposed project uses the datasets that are trained using the count vectorizer method for the detection of fake news and its accuracy will be tested using machine learning algorithms.

Index

Table of Contents

1. INTRODUCTION1

- 1. Machine Learning & NLP1
 - 1.1 Machine Learning1
 - 1.2 Natural Language Processing3
 - 1.2.1 Stages in NLP3
 - 1.2.1 Lexical Analysis3
 - 1.2.1 Syntactic Analysis (Parsing)3
 - 1.2.1 Semantic Analysis3
 - 1.2.1 Discourse Integration4
 - 1.2.1 Pragmatic Analysis4

2. Motivation of work4

3. Problem Statement7

2. LITERATURE SURVEY8

- 1. Introduction8
- 2. Review of Literature8
- 3. Previous Contributions10
- 4. Related Work11
 - 4.1 Spam Detection11
 - 4.2 Stance Detection12

3. METHODOLOGY13

- 1. Proposed System13
- 2. System Architecture13
- 3. Algorithms for proposed system14

4. DATASET15

- 1. Existing Dataset for this system15
- 2. Proposed Dataset Used16
- 3. Fake News Samples17

5. CONCEPTS18

- 1. Pre-processing18
- 2. Steps in text pre-processing18
 - 2.1 Text Normalization18
 - 2.2 Stop Word Removal 18
 - 2.2.1 Stop Word 18

2.3 Stemming	19
2.3.1 Rules of Suffix Stripping Streamers	19
2.3.2 Rules of Suffix Substitution Stemmers	19
3. Count Vectorizer	20
4. Logistic Regression	21
5. Evaluation Measures	21
5.1. Different Types of Evaluation Metrics	22
5.2. Defining the Metrics	23
5.2.1. Accuracy	23
5.2.2. Precision	23
5.2.3. ReCall	23
6. SYSTEM ANALYSIS	24
1. System Configuration	24
1.1. Hardware Requirements	24
1.2. Software Requirements	24
2. Sample Input	25
7. IMPLEMENTATION	28
1. Data Collection and Analysis	28
2. Definition and Details	29
2.1. Pre-processing Data	29
3. Steps	30
3.1. Feature generations	30
4. Vectorizing Data	31
4.1. Bag- of- word	31
4.2. TF-IDF	32
5. Algorithms Used for Classifications	32
5.1. Logistic Regression	33
5.2. Decision Tree Classifier	33
5.3. Random Forest Classifier	34
5.4. Bagging Bootstrap Application	34

5.5. Feature Randomness	35
5.6. Stochastic gradient Descent	36
5.7. Gradient Boosting Algorithm	37
5.8. XGB Classifier	37
5.9. Naive Bayes Classifier	38
6. Implementation Steps	40
6.1. Static Search implementation	40
7. Evaluation Matrices	41
7.1. Random Matrix	42
7.2. Dataset Split using K-fold cross	43
7.3. Confusion Matrix for dynamic System	44
8. FUTURE SCOPE	46
9. CONCLUSION	48
10. REFERENCES	49

1. INTRODUCTION

1.1 MACHINE LEARNING AND NLP:

1.1.1 MACHINE LEARNING Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T , as measured by P , improves with experience E ." This is Alan Turing's definition of machine learning.

Deep learning is a class of machine learning algorithms that utilizes a hierarchical level of artificial neural networks to carry out the process of machine learning. The artificial neural networks are built like the human brain, with neuron nodes connected together like a web. While traditional programs build analysis with data in a linear way, the hierarchical function of deep learning systems enables machines to process data with a nonlinear approach.

The word "deep" in "deep learning" refers to the number of layers through which the data is transformed. More precisely, deep learning systems have a substantial credit assignment path (CAP) depth. The CAP is the chain of transformations from input to output. CAPs describe potentially causal connections between input and output.

For a feedforward neural network, the depth of the CAPs is that of the network and is the number of hidden layers plus one (as the output layer is also parameterized). For recurrent neural networks, in which a signal may propagate through a layer more than once, the CAP depth is potentially unlimited.

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases superior to human experts.

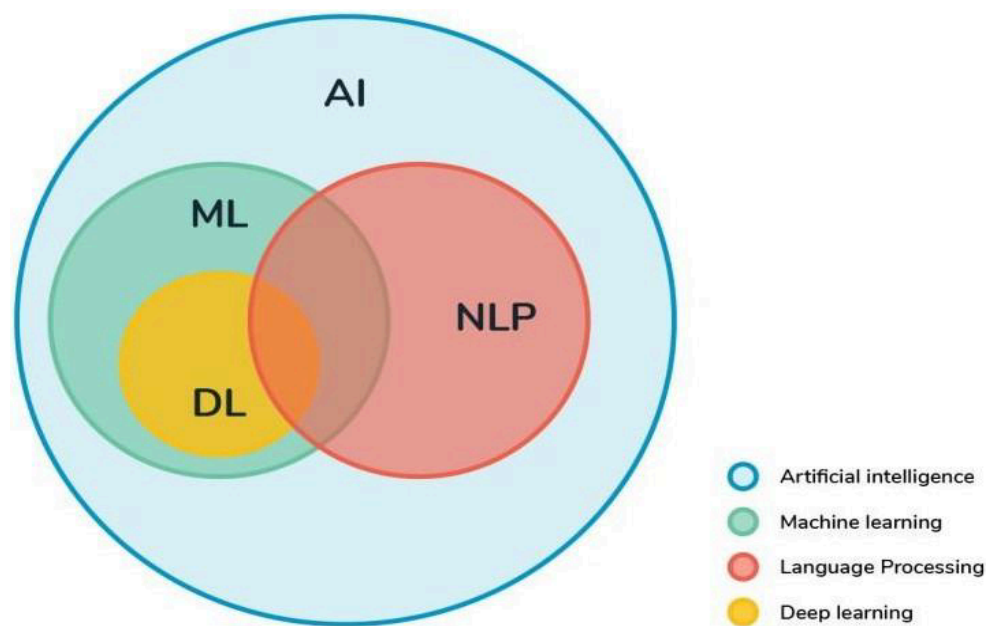


Fig. 1 : Graphical representation of relationship between various fields in artificial intelligence (source: devopedia.org)

1.1.2 NATURAL LANGUAGE PROCESSING

NLP is an area of computer science and artificial intelligence concerned with the interactions between computers and human (natural) languages, how to program computers to fruitfully process large amounts of natural language data.

Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, how to program computers to process and analyze large amounts of natural language data.

1.1.2.1 STAGES IN NLP

1.1.2.1.1 LEXICAL ANALYSIS

Lexical Analysis involves identifying and the structure of words. Lexicon of a language means the collection of words and phrases in a language. Lexical analysis is dividing the whole chunk of text into paragraphs, sentences, and words.

1.1.2.1.2 SYNTACTIC ANALYSIS (PARSING)

Syntactic Analysis involves analysis of words in the sentence for grammar and arranging words in a manner that shows the relationship among the words. The sentence such as “The school goes to boy” is rejected by English syntactic analysis.

1.1.2.1.3 SEMANTIC ANALYSIS

Semantic Analysis draws the exact meaning or the dictionary meaning from the text. The text is checked for meaningfulness. It is done by mapping syntactic structures and objects in the task domain. The semantic analyzer disregards sentence such as “hot ice-cream”

1.1.2.1.4 DISCOURSE INTEGRATION

The meaning of any sentence depends upon the meaning of the sentence just before it. In addition, it also brings about the meaning of immediately succeeding sentences. So in Discourse Integration gives the meaning based on all the sentences given before it. Eg. Consider the sentence “Water is flowing on the bank of the river” But bank has two meanings One Financial Institute and Two River of the bank here System has to consider the second meaning.

1.1.2.1.5 PRAGMATIC ANALYSIS

During this, what was said is re-interpreted on what it actually meant. It involves deriving those aspects of language which require real world knowledge.

1.2 MOTIVATION OF WORK

The rise of fake news during the 2016 U.S. Presidential Election highlighted not only the dangers of the effects of fake news but also the challenges presented when attempting to separate fake news from real news. Fake news may be a relatively new term but it is not necessarily a new phenomenon. Fake news has technically been around at least since the appearance and popularity of one-sided, partisan newspapers in the 19th century. However, advances in technology and the spread of news through different types of media have increased the spread of fake news today. As such, the effects of fake news have increased exponentially in the recent past and something must be done to prevent this from continuing in the future.

I have identified the three most prevalent motivations for writing fake news and chosen only one as the target for this project as a means to narrow the search in a meaningful way.

The first motivation for writing fake news, which dates back to the 19th century one-sided party newspapers, is to influence public opinion. The second, which requires more recent advances in technology, is the use of fake headlines as clickbait to raise money. As such, this paper will focus primarily on fake news as defined by politifact.com, “fabricated content that intentionally masquerades as news coverage of actual events.” This definition excludes satire, which is intended to be humorous ⁸ and not deceptive to readers. Most satirical articles come from sources. Satire can already be classified, by machine learning techniques. Therefore, our goal is to move beyond these achievements and use machine learning to classify, at least as well as humans, more difficult discrepancies between real and fake news.

The dangerous effects of fake news, as previously defined, are made clear by events in which a man attacked a pizzeria due to a widespread fake news article. This story along with analysis provide evidence that humans are not very good at detecting fake news, possibly not better than chance. As such, the question remains whether machines can do a better job.

There are two methods by which machines could attempt to solve the fake news problem better than humans. The first is that machines are better at detecting and keeping track of statistics than humans, for example it is easier for a machine to detect that the majority of verbs used are “suggests” and “implies” versus, “states” and “proves.” Additionally, machines may be more efficient in surveying a knowledge base to find all relevant articles and answering based on those many different sources. Either of these methods could prove useful in detecting fake news, but we decided to focus on how a machine can solve the fake news problem using supervised learning that extracts feature of the language and content only within the source in question, without utilizing any fact checker or knowledge base. For many fake news detection techniques, a “fake” article published by a trustworthy author through a trustworthy source would not be caught. This approach would combat those “false negative” classifications of fake news. In essence, the task would be equivalent to what a human face when reading a hard copy of a newspaper article, without internet access or outside knowledge of the subject (versus reading something online where he can simply look up relevant sources). The machine, like the human in the coffee shop, will have only access to the words in the article and must use strategies that do not rely on blacklists of authors.

The current project involves utilizing machine learning and natural language processing techniques to create a model that can expose documents that are, with 9 high probability, fake news articles. Many of the current automated approaches to this problem are centred around a “blacklist” of authors and sources that are known producers of fake news. But, what about when the author is unknown or when fake news is published through a generally reliable source? In these cases, it is necessary to rely simply on the content of the news article to make a decision on whether or not it is fake. By collecting examples of both real and fake news and training a model, it should be possible to classify fake news articles with a certain degree of accuracy. The goal of this project is to find the effectiveness and limitations of language-based techniques for detection of fake news using machine learning algorithms including but not limited to convolutional neural networks and recurrent neural networks. The outcome of this project should determine how much can be achieved in this task by analysing patterns contained in the text and blind to outside information about the world.

1.3 PROBLEM STATEMENT

News consumption is a double-edged sword. On the one hand, its low cost, easy access, and rapid dissemination of information lead people to seek out and consume news. It enables the wide spread of “fake news”, i.e., low quality news with intentionally false information. The extensive spread of fake news has the potential for extremely negative impacts on individuals and society. Therefore, fake news detection has recently become emerging research that is attracting tremendous attention. First, fake news is intentionally written to mislead readers to believe false information, which makes it difficult and nontrivial to detect based on news content.

To develop a **FAKE NEWS DETECTION** system using natural language processing and its accuracy will be tested using machine learning algorithms. The algorithm must be able to detect fake news in each scenario.

2. LITERATURE SURVEY

2.1 INTRODUCTION

In the world of rapidly increasing technology, information sharing has become an easy task. There is no doubt that the internet has made our lives easier and access to lots of information. This is an evolution in human history, but at the same time it unfocussed the line between true media and maliciously forged media. Today anyone can publish content – credible or not – that can be consumed by the world wide web. Sadly, fake news accumulates a great deal of attention over the internet, especially on social media. People get deceived and don't think twice before circulating such mis-informative pieces to the world. This kind of news vanishes but not without doing the harm it intended to cause. The social media sites like Facebook, Twitter, WhatsApp play a major role in supplying this false news. Many scientists believe that counterfeited news issues may be addressed by means of machine learning and artificial intelligence.

Various models are used to provide an accuracy range of 60-75%. Which comprises Naive Bayes classifier, Linguistic features based, Bounded decision tree model, SVM etc. The parameters that are taken in consideration do not yield high accuracy. The motive of this project is to increase the accuracy of detecting fake news more than the present results that are available. By fabricating this new model which will judge the counterfeit news articles on the basis of certain criteria like spelling mistakes, jumbled sentences, punctuation errors, words used.

2.2 REVIEW OF LITERATURE

There are two categories of important research in automatic classification of real and fake news up to now:

In the first category, approaches are at conceptual level, distinction among fake news is done for three types: serious lies (which means news is about wrong and unreal events or

information like famous rumors), tricks (e.g., providing wrong information) and comics (e.g., funny news which is an imitation of real news but contain bizarre contents).

In the second category, linguistic approaches and reality considerations techniques are used at a practical level to compare the real and fake contents. Linguistic approaches try to detect text features like writing styles and contents that can help in distinguishing fake news. The main idea behind this technique is that linguistic behaviors like using marks, choosing various types of words or adding labels for parts of a lecture are rather unintentional, so they are beyond the author's attention. Therefore, an appropriate intuition and evaluation of using linguistic techniques can reveal hoping results in detecting fake news.[1]

Robbins studied the distinction between the contents of real and comic news via multilingual features, based on a part of comparative news (The Onion, and The Beaverton) and real news (The Toronto Star and The New York Times) in four areas of civil, science, trade, and ordinary news. She obtained the best performance of detecting fake news with a set of features including unrelated, marking and grammar.[10]

Bondelli believes that the cooperation of information technology specialists in reducing fake news is very important. In order to deal with fake news, using data mining as one of the techniques has attracted many researchers. In data mining-based approaches, data integration is used in detecting fake news. In the current business world, data are an ever-increasing valuable asset, and it is necessary to protect sensitive information from unauthorized people. However, the prevalence of content publishers who are willing to use fake news leads to ignoring such endeavors. Organizations have invested a lot of resources to find effective solutions for dealing with clickbait effects. [7]

2.3 PREVIOUS CONTRIBUTIONS

Shloka Gilda presented a concept approximately how NLP is relevant to stumble on fake information. They have used time period frequency-inverse record frequency (TFIDF) of bi-grams and probabilistic context free grammar (PCFG) detection. They have examined their dataset over more than one class algorithms to find out the great model. They locate that TF-IDF of bi-grams fed right into a stochastic gradient descent model identifies non-credible resources with an accuracy of 77%.[2]

Mykhailo Granik proposed a simple technique for fake news detection: the usage of naïve Bayes classifiers. They used BuzzFeed news forgetting to know and trying out the naïve Bayes classifier. The dataset is taken from Facebook news publish and completed accuracy up to 74% on test set.[4]

Cody Buntain advanced a method for automating fake news detection on twitter. They applied this method to twitter content sourced from BuzzFeed's fake news Dataset. Furthermore, leveraging non-professional, crowdsourced people instead of Journalists presents a beneficial and much less costly way to classify proper and fake Memories on twitter rapidly.[3]

Zang offered a paper which allows us to recognize how social networks and gadget studying (ML) strategies may be used for faux news detection. They have used novel ML fake news detection method and carried out this approach inside a Facebook Messenger chatbot and established it with a actual-world application, acquiring a fake information detection accuracy of 81%. It aims to present an insight of characterization of news stories in the modern diaspora combined with the differential content types of news stories and its impact on readers. Subsequently, we dive into existing fake news detection approaches that are heavily based on text- based analysis, and also describe popular fake news datasets. We conclude the paper by identifying 4 key open research challenges that can guide future research. It is a theoretical Approach which gives Illustrations of fake news detection by analyzing the psychological factors.[9]

Hokmabadi et. al. gave a framework based on a different machine learning approach that deals with various problems including accuracy shortage, time lag (BotMaker) and high processing time to handle thousands of tweets in 1 sec. Firstly, they have collected 400,000 tweets from the HSpam14 dataset. Then they further characterize the 150,000 spam tweets and 250,000 non-spam tweets. They also derived some lightweight features along with the Top-30 words that are providing the highest information gain from the Bag-of-Words model. 4. They were able to achieve an accuracy of 91.65% and surpassed the existing solution by approximately 18%. [12]

2.4 RELATED WORK

2.4.1 SPAM DETECTION

The problem of detecting not-genuine sources of information through content-based analysis is considered solvable at least in the domain of spam detection [7], spam detection utilizes statistical machine learning techniques to classify text (i.e., tweets [8] or emails) as spam or legitimate. These techniques involve pre-processing of the text, feature extraction (i.e., bag of words), and feature selection based on which features lead to the best performance on a test dataset. Once these features are obtained, they can be classified using Naive Bayes, Support Vector Machines, TF-IDF, or K-nearest neighbors' classifiers. All of these classifiers are

$$f(message, \theta) = \begin{cases} C_{\text{spam}} & \text{if classified as spam} \\ C_{\text{leg}} & \text{otherwise} \end{cases}$$

characteristic of supervised machine learning, meaning that they require some labeled data in order to learn the function where, m is the message to be classified and is a vector of parameters and C_{spam} and C_{leg} are respectively spam and legitimate messages. The task of detecting fake news is similar and almost analogous to the task of spam detection in that both aim to separate examples of legitimate text from examples of illegitimate, ill-intended texts.

2.4.2 STANCE DETECTION

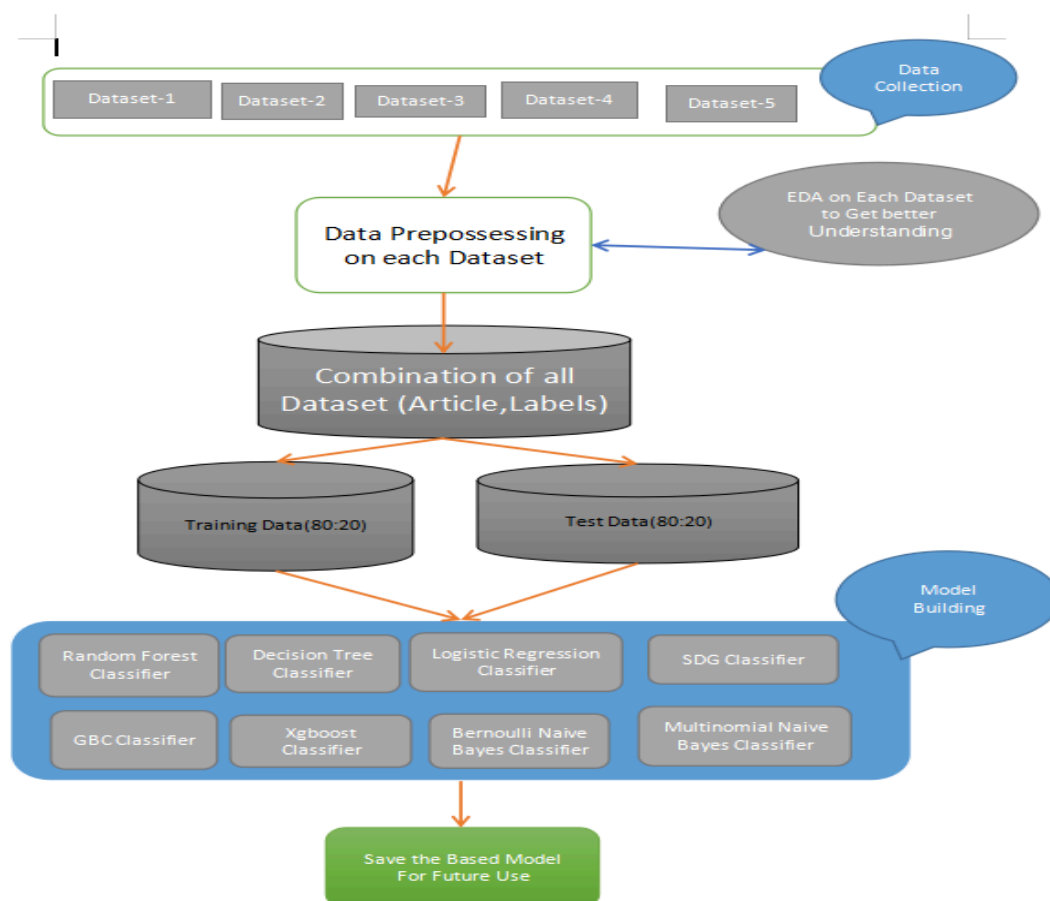
The goal of this contest was to encourage the development of tools that may help human fact checkers identify deliberate misinformation in news stories through the use of machine learning, natural language processing and artificial intelligence. The organizers decided that the first step in this overarching goal was understanding what other news organizations are saying about the topic in question. As such, they decided that stage one of their contest would be a stance detection competition. More specifically, the organizers built a dataset of headlines and bodies of text and challenged competitors to build classifiers that could correctly label the stance of a body text, relative to a given headline, into one of four categories: “agree”, “disagree”, “discusses” or “unrelated.” The top three teams all reached over 80% accuracy on the test set for this task. The top teams model was based on a weighted average between gradient-boosted decision trees and a deep convolutional neural network.

3. METHODOLOGY

3.1 PROPOSED SYSTEM

The proposed system when subjected to a scenario of a set of news articles, the new articles are categorized as true or fake by the existing data available. This prediction is done by using the relationship between the words used in the article with one another. The proposed system contains a Word2Vec model for finding the relationship between the words and with the obtained information of the existing relations, the new articles are categorized into fake and real news.

3.2 SYSTEM ARCHITECTURE



Input is collected from various sources such as newspapers, social media and stored in datasets. System will take input from datasets. The datasets undergo preprocessing, and the unnecessary information is removed from it and the data types of the columns are changed if required. Jupiter notebook and python libraries are used in the above step. Count vectorizer technique is used in the initial step.

For fake news detection

- We must train the system using a dataset. Before entering to the detection of fake news
- The entire dataset is divided into two datasets. 80% is used for training and 20% is used for testing. During training, the Logistic Regression algorithm is used to train the model using the train dataset. In testing, the test dataset is given as input and the output is predicted. After the testing time, the predicted output and the actual output are compared using confusion matrix obtained. The confusion matrix gives the information regarding the number of correct and wrong predictions in the case of real and fake news.

The accuracy is calculated by the equation $\text{No of Correct Predictions} / \text{Total Test}$

Dataset Input Size

3.3 ALGORITHM FOR THE PROPOSED SYSTEM:

Step 1: Start

Step 2: Input is collected from various sources and prepared a dataset.

Step 3: Preprocessing of data is done, and the dataset is divided into 2 parts: training and testing data.

Step 4: Vectorization techniques are used to convert the train data into numerical.

Step 5: Logistic Regression algorithm is used to build the predictive model using the train data

Step 6: Accuracy is calculated.

4. DATASETS

4.1 EXISTING DATASETS FOR THIS SYSTEM:

The lack of manually labeled fake news datasets is certainly a bottleneck for advancing computationally intensive, text-based models that cover a wide array of topics. The dataset for the fake news challenge does not suit our purpose due to the fact that it contains the ground truth regarding the relationships between texts but not whether or not those texts are actually true or false statements. For our purpose, we need a set of news articles that is directly classified into categories of news types (i.e. real vs. fake or real vs parody vs. clickbait vs. propaganda). For more simple and common NLP classification tasks, such as sentiment analysis, there is an abundance of labeled data from a variety of sources including Twitter, Amazon Reviews, and IMDb Reviews. Unfortunately, the same is not true for finding labeled articles of fake and real news. This presents a challenge to researchers and data scientists who want to explore the topic by implementing supervised machine learning techniques. I have researched the available datasets for sentence-level classification and ways to combine datasets to create full sets with positive and negative examples for document-level classification.

4.2 PROPOSED DATASET USED:

There exists no dataset of similar quality to the Dataset for document level classification of fake news. As such, we had the option of using the headlines of documents as statements or creating a hybrid dataset of labeled fake and legitimate news articles. This shows an informal and exploratory analysis carried out by combining two datasets that individually contain positive and negative fake news examples. Genes trains a model on a specific subset of both the Kaggle dataset and the data from NYT and the Guardian.

- Dataset 1

```
Dataset1.head()
```

	Unnamed: 0		title	text	label
0	8476		You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE
1	10294		Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg LinkedIn Reddit Stumbleu...	FAKE
2	3608		Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL
3	10142		Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE
4	875		The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL

- Dataset 2

```
Dataset2_fake.head()
```

		title	text	subject	date
0		Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn't wish all Americans ...	News	December 31, 2017
1		Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017
2		Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017
3		Trump Is So Obsessed He Even Has Obama's Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017
4		Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017

- Dataset 3

```
Dataset3_real.head()
```

	id	news_url	title	tweet_ids
0	politifact14984	http://www.nfib-sbet.org/	National Federation of Independent Business	967132259869487105t967164368768196609t967215...
1	politifact12944	http://www.cq.com/doc/newsmakertranscripts-494...	comments in Fayetteville NC	942953459t8980098198t16253717352t1668513250...
2	politifact333	https://web.archive.org/web/20080204072132/htt...	Romney makes pitch, hoping to close deal : Ele...	NaN
3	politifact4358	https://web.archive.org/web/20110811143753/htt...	Democratic Leaders Say House Democrats Are Uni...	NaN
4	politifact779	https://web.archive.org/web/20070820164107/htt...	Budget of the United States Government, FY 2008	89804710374154240t91270460595109888t96039619...

- Dataset 4

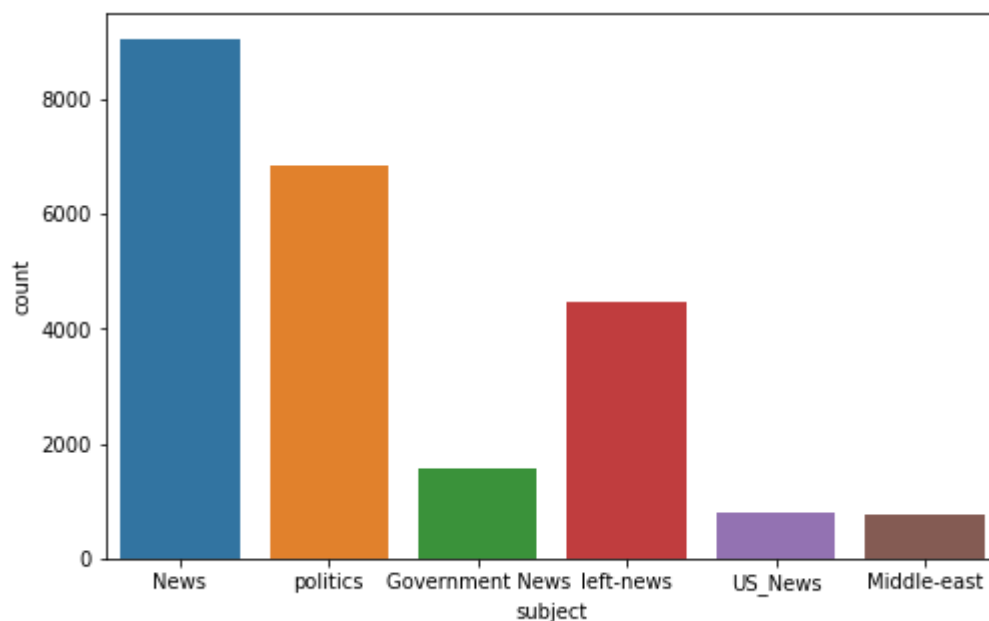
Dataset4.head()

	id	title	author	text	label
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's Let...	1
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1
3	3	15 Civilians Killed In Single US Airstrike Hav...	Jessica Purkiss	Videos 15 Civilians Killed In Single US Aistr...	1
4	4	Iranian woman jailed for fictional unpublished...	Howard Portnoy	Print \nAn Iranian woman has been sentenced to...	1

- Dataset 5

Dataset5.head()

	URLs	Headline	Body	Label
0	http://www.bbc.com/news/world-us-canada-414191...	Four ways Bob Corker skewered Donald Trump	Image copyright Getty Images\nOn Sunday mornin...	1
1	https://www.reuters.com/article/us-filmfestiva...	Linklater's war veteran comedy speaks to moder...	LONDON (Reuters) - "Last Flag Flying", a comed...	1
2	https://www.nytimes.com/2017/10/09/us/politics...	Trump's Fight With Corker Jeopardizes His Legi...	The feud broke into public view last week when...	1
3	https://www.reuters.com/article/us-mexico-oil-...	Egypt's Cheiron wins tie-up with Pemex for Mex...	MEXICO CITY (Reuters) - Egypt's Cheiron Holdin...	1
4	http://www.cnn.com/videos/cnnmoney/2017/10/08/...	Jason Aldean opens 'SNL' with Vegas tribute	Country singer Jason Aldean, who was performin...	1



Categories of News present in Datasets

In his experiment, the topics involved in training and testing are restricted to U.S News, Politics, Business and World news. However, he does not account for the difference in date range between the two datasets, which likely adds an additional layer of topic bias based on

topics that are popular during specific periods of time. We have collected data in a manner like that of Genes, but more cautious in that we control for more bias in the sources and topics. Because the goal of our project was to find patterns in the language that are indicative of real or fake news, having source bias would be detrimental to our purpose. Including any source bias in our dataset, i.e., patterns that are specific to NYT, The Guardian, or any of the fake news websites, would allow the model to learn to associate sources with real/fake news labels. Learning to classify sources as fake or real news is an easy problem but learning to classify specific types of language and language patterns as fake or real news is not.

5. CONCEPTS

5.1 PREPROCESSING:

In any Machine Learning process, Data Pre-processing is that step in which the data gets transformed, or encoded, to bring it to such a state that now the machine can easily parse it. In other words, the features of the data can now be easily interpreted by the algorithm.

In this fake news detection, pre-processing is the major thing that should be done Firstly as the data dataset is collected from various sources unnecessary information should be removed, converted to lowercase, remove punctuation, symbols, stop words.

5.2 STEPS IN TEXT PRE-PROCESSING:

5.2.1 TEXT NORMALIZATION:

Text normalization is a process of transforming text into a single canonical form. Normalizing text before storing or processing it allows for separation of required data from the rest so that the system can send consistent data as an input to the other steps of the algorithm.

5.2.2 STOP WORD REMOVAL

5.2.2.1 Stop Word:

A Stop Word is a commonly used word in any natural language such as “a, an , the, for, is, was, which, are, were, from, do, with, and, so, very, that, this, no, yourselves etc....”.

These Stop Words will have a very high frequency and so these should be eliminated while calculating the term frequency so that the other important things are given priority. Stop word removal is such a Pre-processing step which removes these stop words and thereby helping in the further steps and also reducing some processing time because the size of the document decreases tremendously.

Consider a Sentence

“This is a sample sentence, showing off the stop word removal”.

Output after Stop word removal is:

[“sample”, “sentence”, “showing”, “stop”, “word”, “removal”]

Note: Though Stop words refer to the most commonly used words in a particular language, there is no single universal list of stop words, different tools use different stop words.

5.2.3 STEMMING:

Stemming is a pre-processing step in Text Mining applications as well as a very common requirement of Natural Language processing functions. In fact it is very important in most of the Information Retrieval systems. The main purpose of stemming is to reduce different grammatical forms / word forms of a word like its noun, adjective, verb, adverb etc. to its root form. The goal of stemming is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form.

Eg: A stemmer for English should identify the strings **"cats"**, **"catlike"**, **"catty"** as based

on the root "cat".

5.2.3.1 RULES OF SUFFIX STRIPPING STEMMERS:

1.If the word ends in 'ed', remove the 'ed'.

2.If the word ends in 'ing', remove the 'ing'.

3.If the word ends in 'ly', remove the 'ly'.

5.2.3.2 RULES OF SUFFIX SUBSTITUTION STEMMERS:

1.If the word ends in 'ies' substitute 'ies' with 'y'.

Generally this stemmer is used because of some words like families etc...

5.3 COUNTVECTORIZER:

Count Vectorizer tokenizer (tokenization means breaking down a sentence or paragraph or any text into words) the text along with performing very basic preprocessing like removing the punctuation marks, converting all the words to lowercase, etc.The vocabulary of known

Words are formed which are also used for encoding unseen text later.An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document.

5.3.1 Input to count vectorizer:

Document having 3

sentences sam

sam is super happy

sam sam is very sad

sam sam is scary angry

	angry	happy	is	sad	sam	scary	super	very
0	0	1	1	0	2	0	1	0
1	0	0	1	1	2	0	0	1
2	1	0	1	0	2	1	0	0

5.4 Logistic Regression:

It is a classification not a regression algorithm. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on a given set of independent variables(s). In simple words, it predicts the probability of occurrence of an event by fitting data to a logit

function. Hence, it is also known as logit regression. Since it predicts the probability, its output value lies between 0 and 1 (as expected).

Mathematically, the log odds of the outcome are modelled as a linear combination of the predictor variables.

Odds = $p/(1-p)$ = probability of event occurrence / probability of not event occurrence

$\ln(\text{odds}) = \ln(p/(1-p))$

$\text{logit}(p) = \ln(p/(1-p)) = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + \dots + b_kX_k$.

5.5 EVALUATION MEASURES:

Whenever we build Machine Learning models, we need some form of metric to measure the goodness of the model. Bear in mind that the “goodness” of the model could have multiple interpretations, but generally when we speak of it in a Machine Learning context we are talking of the measure of a model's performance on new instances that weren't a part of the training data.

Determining whether the model being used for a specific task is successful depends on 2 key factors:

1. Whether the evaluation metric we have selected is the correct one for our problem
2. If we are following the correct evaluation process

In this article, I will focus only on the first factor — Selecting the correct evaluation metric.

5.5.1 DIFFERENT TYPES OF EVALUATION METRICS

The evaluation metric we decide to use depends on the type of NLP task that we are doing. To further add, the stage the project is at also affects the evaluation metric we are using. For instance, during the model building and deployment phase, we'd more often than not use a different evaluation metric to when the model is in production. In the first 2 scenarios, ML metrics would suffice but in production, we care about business impact, therefore we'd rather use business metrics to measure the goodness of our model.

With that being said, we could categorize evaluation metrics into 2 buckets.

- Intrinsic Evaluation — Focuses on intermediary objectives (i.e. the performance of an NLP component on a defined subtask)

- Extrinsic Evaluation — Focuses on the performance of the final objective (i.e. the performance of the component on the complete application)

Stakeholders typically care about extrinsic evaluation since they'd want to know how good the model is at solving the business problem at hand. However, it's still important to have intrinsic evaluation metrics in order for the AI team to measure how they are doing. We will be focusing more on intrinsic metrics for the remainder of this article.

5.5.2 DEFINING THE METRICS

Some common intrinsic metrics to evaluate NLP systems are as follows:

5.5.2.1 ACCURACY

Whenever the accuracy metric is used, we aim to learn the closeness of a measured value to a known value. It's therefore typically used in instances where the output variable is categorical or discrete — Namely a classification task.

5.5.2.2 PRECISION

In instances where we are concerned with how exact the model's predictions are we would use Precision. The precision metric would inform us of the number of labels that are actually labeled as positive in correspondence to the instances that the classifier labeled as positive.

5.5.2.3 RECALL

Recall measures how well the model can recall the positive class (i.e. the number of positive labels that the model identified as positive)

6. SYSTEM ANALYSIS

6.1 SYSTEM CONFIGURATION

This project can run on commodity hardware. We ran the entire project on an Intel I5 processor with 8 GB Ram, 2 GB Nvidia Graphic Processor. It also has 2 cores which run at 1.7 GHz, 2.1 GHz respectively. First part is the training phase which takes 10-15 mins of time and the second part is the testing part which only takes a few seconds to make predictions and calculate accuracy.

6.1.1 HARDWARE REQUIREMENTS:

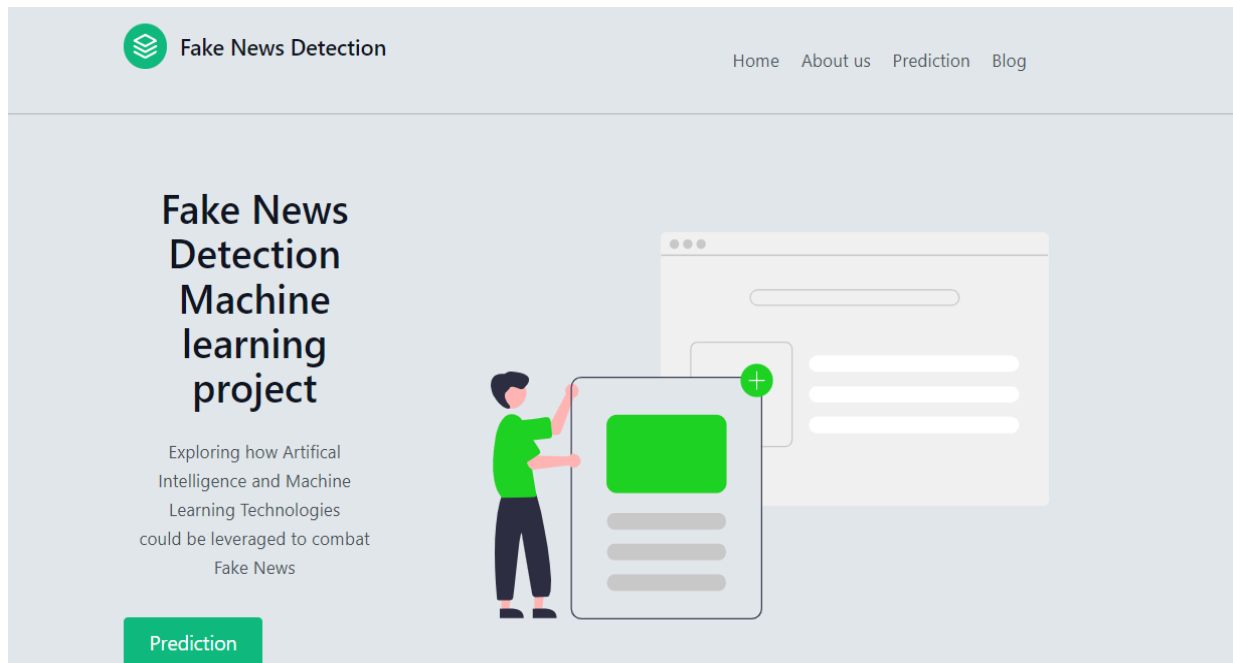
- RAM: 4 GB
- Storage: 500 GB
- CPU: 2 GHz or faster
- Architecture: 32-bit or 64-bit

6.1.2 SOFTWARE REQUIREMENTS

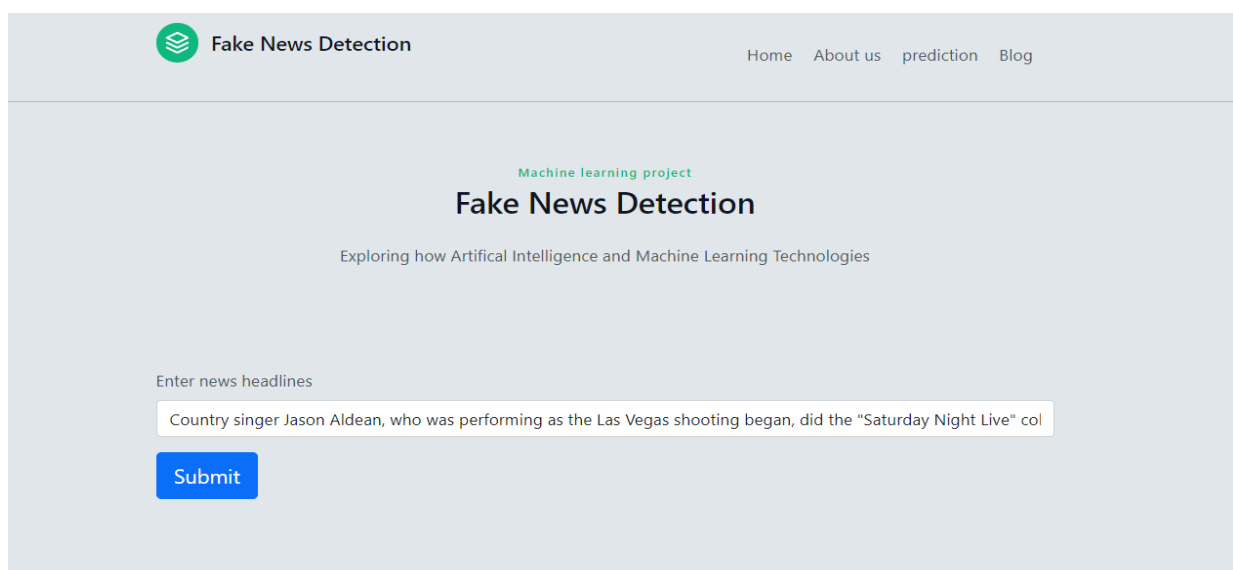
- Python 3.5 in Google Collab is used for data pre-processing, model training and prediction.
- Operating System: Windows 7 and above or Linux based OS or MAC OS.

6.2 Sample Input

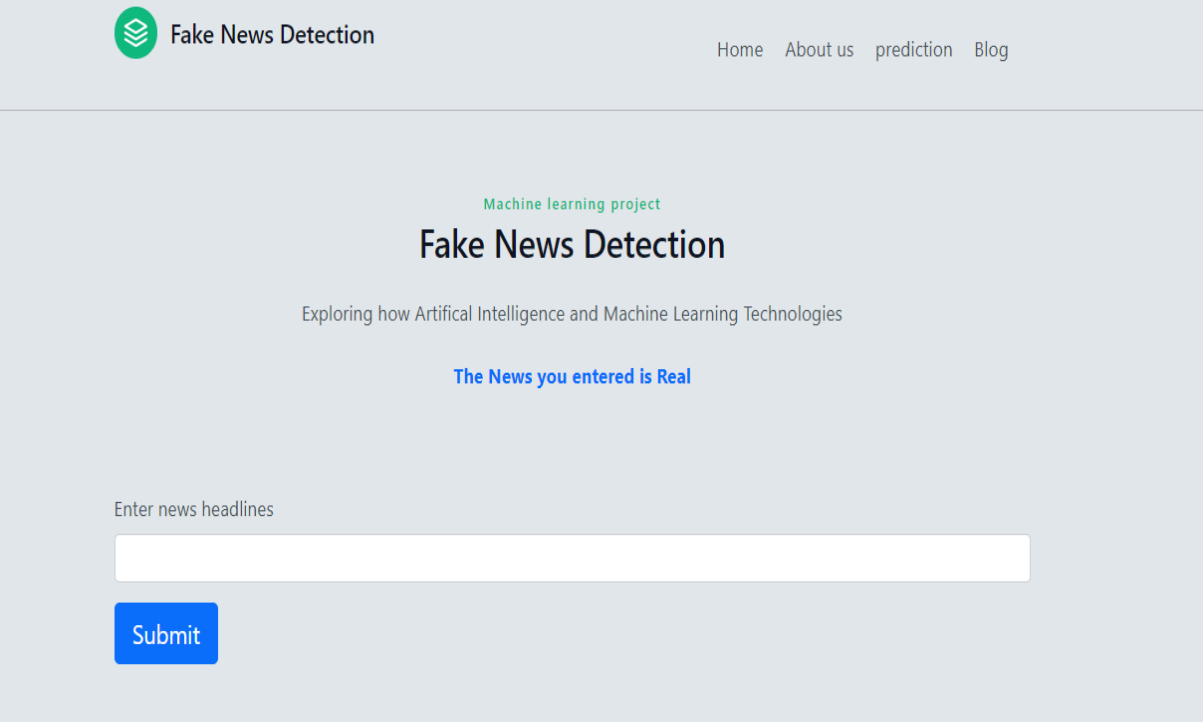
- Open Fake News Detection Web Interface



- Click on Prediction Tab and insert the news in Text field and click on Submit.



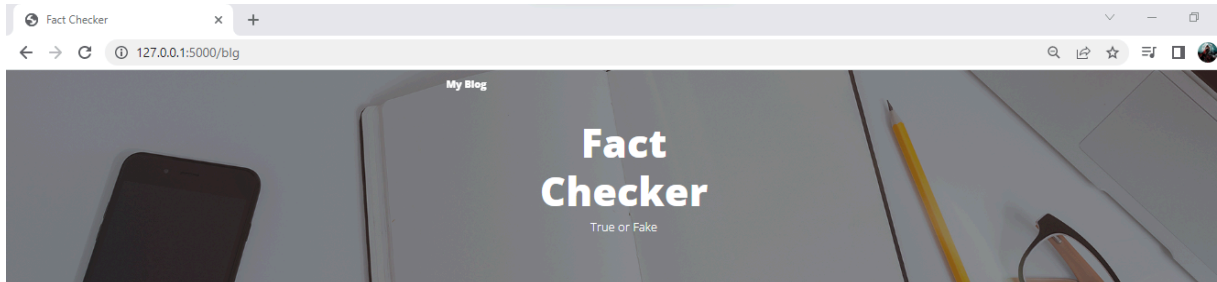
- Check the result of the News. Fake/Real



The screenshot shows the homepage of a web application titled "Fake News Detection". At the top, there is a navigation bar with a logo (a green circle with three white stacked lines) and the text "Fake News Detection". To the right of the logo are links: "Home", "About us", "prediction", and "Blog". Below the navigation bar, the main content area has a light gray background. It features the text "Machine learning project" in green, followed by the title "Fake News Detection" in large black font. Below the title is the subtitle "Exploring how Artificial Intelligence and Machine Learning Technologies". A blue text message "The News you entered is Real" is displayed. At the bottom, there is a text input field with the placeholder "Enter news headlines" and a blue "Submit" button.

- **Extra Feature:** Accessing the Blog for Detecting Real Time News (Fake/True)

Step 1: Click on the Blog option of Web Page



**Man must explore,
and this is
exploration at its
greatest**

Posted by Kery man on September 24, 2022

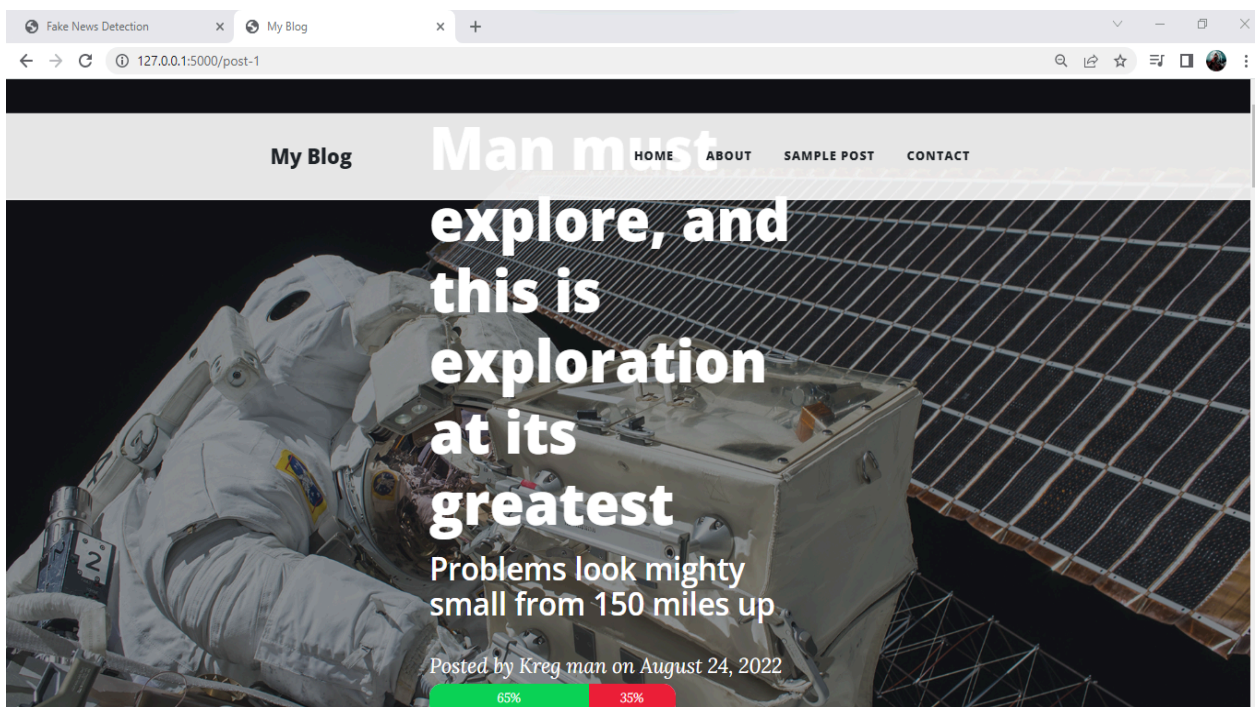
**I believe every
human has a finite
number of
heartbeats. I don't
intend to waste any
of mine.**

Posted by Aurthur cheng on September 18, 2022

Science has not yet

Step 2: Click on the News

Step 3: Check the Poll Result(Fake/True)



7. IMPLEMENTATION

7.1 DATA COLLECTION AND ANALYSIS

We can get online news from different sources like social media websites, search engines, homepage of news agency websites or fact-checking websites. On the Internet, there are a few publicly available datasets for Fake news classification like Buzzfeed News, LIAR, BS Detector etc. These datasets have been widely used in different research papers for determining the veracity of news. In the Following sections, I have discussed in brief about the sources of the dataset used in this work.

Online news can be collected from different sources, such as news agency homepages, search engines, and social media websites. However, manually determining the veracity of news is a challenging task, usually requiring annotators with domain expertise who performs careful analysis of claims and additional evidence, context, and reports from authoritative sources. Generally, news data with annotations can be gathered in the following ways: Expert journalists, Fact-checking websites, Industry detectors, and Crowdsourced workers. However, there are no agreed upon benchmark datasets for the fake news detection problem. Data gathered must be pre-processed- that is, cleaned, transformed, and integrated before it can undergo the training process. The dataset that we used is explained below:

A. PolitiFact Dataset:

This dataset is collected from fact-checking website PolitiFact through its API. It includes 12,836 human labelled short statements, which are sampled from various contexts, such as news releases, TV or radio interviews, campaign speeches, etc. The labels for news truthfulness are fine-grained multiple classes: pants-fire, false, barely true, half-true, mostly true, and true. The data source used for this project is the PolitiFact dataset which contains 3 files with .csv format for test, train and validation. Below is some description about the data files used for this project.

A.1. **PolitiFact**: A Benchmark Dataset for Fake News Detection sourced from an Online Community of Datascientist and Machine Learning Practitioners known as Kaggle.

- Column1: Statement (News headline or text).
- Column2: Label (Label class contains: True, False)

The dataset used for this project were in csv format named train.csv, test.csv and valid.csv.

2. REAL_OR_FAKE.CSV we used this dataset for passive aggressive classifiers. It contains 3 columns viz 1- Text/keyword, 2-Statement,

3-Label (Fake/True)

7.2 DEFINITIONS AND DETAILS

A. Pre-processing Data

Social media data is highly unstructured – majority of them are informal communication with typos, slangs and bad-grammar etc. Quest for increased performance and reliability has made it imperative to develop techniques for utilization of resources to make informed decisions. To achieve better insights, it is necessary to clean the data before it can be used for predictive modeling. For this purpose, basic pre-processing was done on the News training data. This step was comprised of- Data Cleaning:

While reading data, we get data in the structured or unstructured format. A structured format has a well-defined pattern whereas unstructured data has no proper structure. In between the 2 structures, we have a semi-structured format which is comparably better structured than unstructured format. Cleaning up the text data is necessary to highlight attributes that we're going to want our machine learning system to pick up on. Cleaning (or pre-processing) the data typically consists of a number of

7.3 Steps:

a) Remove punctuation

Punctuation can provide grammatical context to a sentence which supports our understanding. But for our vectorizer which counts the number of words and not the context, it does not add value, so we remove all special characters. eg: How are you?->How are you

b) Tokenization

Tokenizing separates text into units such as sentences or words. It gives structure to previously unstructured text. eg: Plata o Plomo-> 'Plata','o','plomo'.

c) Remove stopwords

Stopwords are common words that will likely appear in any text. They don't tell us much about our data so we remove them. eg: silver or lead is fine for me-> silver, lead, fine.

d) Stemming

Stemming helps reduce a word to its stem form. It often makes sense to treat related words in the same way. It removes suffices, like

“ing”, “ly”, “s”, etc. by a simple rule-based approach. It reduces the corpus of words but often the actual words get neglected. eg:

Entitling, Entitled -> Entitle. Note: Some search engines treat words with the same stem as synonyms [18].

A. Feature Generation

We can use text data to generate a number of features like word count, frequency of large words, frequency of unique words, n-grams etc. By creating a representation of words that

capture their meanings, semantic relationships, and numerous types of context they are used in, we can enable computers to understand text and perform Clustering, Classification etc [19].

7.4 Vectorizing Data:

Vectorizing is the process of encoding text as integers i.e. numeric form to create feature vectors so that machine learning algorithms can understand our data.

1. Vectorizing Data: Bag-Of-Words

Bag of Words (BoW) or CountVectorizer describes the presence of words within the text data. It gives a result of 1 if present in the sentence and 0 if not present. It, therefore, creates a bag of words with a document-matrix count in each text document.

1.1 Vectorizing Data: N-Grams are simply all combinations of adjacent words or letters of length n that we can find in our source text. Ngrams with $n=1$ are

called unigrams. Similarly, bigrams ($n=2$), trigrams ($n=3$) and so on can also be used. Unigrams usually don't contain much information as compared to bigrams and trigrams. The basic principle behind n -grams is that they capture the letter or word that is likely to follow the given word. The longer the n -gram (higher n), the more context you have to work with [20].

2. Vectorizing Data: TF-IDF

It computes the “relative frequency” that a word appears in a document compared to its frequency across all documents TF-IDF weight represents the relative importance of a term in the document and entire corpus [17]. TF stands for Term Frequency: It calculates how frequently a term appears in a document. Since, every document size varies, a term may appear more in a long-sized document than a short one. Thus, the length of the document often divides Term frequency. Note: Used for search engine scoring, text summarization, document clustering.

IDF stands for Inverse Document Frequency: A word is not of much use if it is present in all the documents. Certain terms like “a”, “an”, “the”, “on”, “of” etc. appear many times in a document but are of little importance. IDF weighs down the importance of these terms and increases the importance of rare ones. The more the value of IDF, the more unique is the word [17]. TF-IDF is applied on the body text, so the relative count of each word in the sentences is stored in the document matrix.

Note: Vectorizers output sparse matrices. Sparse Matrix is a matrix in which most entries are 0 [21].

7.5 Algorithms used for Classification

This section deals with training the classifier. Different classifiers were investigated to predict the class of the text. We explored

specifically, four different machine-learning algorithms – Multinomial Naïve Bayes, Logistic regression, XGBClassifier, Decision Tree Classifier, Random Forest, Stochastic Gradient Descent, Gradient Boosting Algorithm.

The implementations of these classifiers were done using Python library Sci-Kit Learn.

For Training of Dataset, we have experimented with almost all the Machine Learning Algorithms which are practically used for prediction purposes.

Algorithms that we used in this Projects are:

1) Logistic Regression:

It is a classification not a regression algorithm. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given a set of independent variable(s). In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as logit regression. Since it predicts the probability, its output value lies between 0 and 1 (as expected).

Mathematically, the log odds of the outcome are modeled as a linear combination of the predictor variables.

Odds = $p/(1-p)$ = probability of event occurrence / probability of not event

occurrence

$\ln(\text{odds}) = \ln(p/(1-p))$

$\text{logit}(p) = \ln(p/(1-p)) = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + \dots + b_kX_k$.

Accuracy: 87.04 %

Snippet:

```
#LogisticRegression
pipe = Pipeline([('vect', CountVectorizer()),
                 ('tfidf', TfidfTransformer()),
                 ('model', LogisticRegression())])

Logisticmodel = pipe.fit(x_train, y_train)
prediction = Logisticmodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
Logisticmodel_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

accuracy: 87.04%

2) Decision Tree Classifier:

_Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split.

%Accuracy: 82.07 %

Snippet:

```
#####DecisionTreeClassifier
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', DecisionTreeClassifier(criterion= 'entropy',
                                                    max_depth = 10,
                                                    splitter='best',
                                                    random_state=2020))])

DecisionTreemodel = pipe.fit(x_train, y_train)
prediction = DecisionTreemodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
DecisionTreemodel_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

accuracy: 82.07%

3) Random Forest Classifier:

Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we've collection of decision trees (so known as "Forest"). To classify a new object based on attributes, each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest). The random forest is a classification algorithm consisting of many decision trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. The reason that the random forest model works so well is: A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models. So how does random forest ensure that the behavior of each individual tree is not too correlated with the behavior of any of the other trees in the model?

It uses the following two methods:

4. Bagging (Bootstrap Aggregation) — Decisions trees are very sensitive to the data they are trained on — small changes to the training sets can result in significantly different tree structures. Random forest takes advantage of this by allowing each individual tree to

randomly sample from the dataset with replacement, resulting in different trees. This process is known as bagging or bootstrapping.

5) Feature Randomness — In a normal decision tree, when it is time to split a node, we consider every possible feature and pick the one that produces the most separation between the observations in the left node vs. those in the right node. In contrast, each tree in a random

forest can pick only from a random subset of features. This forces even more variation amongst the trees in the model and ultimately results in lower correlation across trees and more diversification [22]

Accuracy: 82.49 %

Snippet:

```
#####RandomForestClassifier
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', RandomForestClassifier())])

RandomForestmodel = pipe.fit(x_train, y_train)
prediction = RandomForestmodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
RandomForestmodel_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

accuracy: 82.49%

6) Stochastic Gradient Descent:

The word ‘*stochastic*’ means a system or a process that is linked with a random probability. Hence, in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration. In Gradient Descent, there is a term called “batch” which denotes the total number of samples from a dataset that is used for calculating the gradient for each iteration. In typical Gradient Descent optimization, like Batch Gradient Descent, the batch is taken to be the whole dataset. Although, using the whole dataset is really useful for getting to the minima in a less noisy and less random manner, the problem arises when our datasets get big.

Suppose you have a million samples in your dataset, so if you use a typical Gradient Descent optimization technique, you will have to use all of the one million samples for completing one iteration while performing the Gradient Descent, and it has to be done for every iteration until the minima are reached. Hence, it becomes computationally very expensive to perform.

This problem is solved by Stochastic Gradient Descent. In SGD, it uses only a single sample, i.e., a batch size of one, to perform each iteration. The sample is randomly shuffled and selected for performing the iteration.

Accuracy: 86.23 %

Snippet:

```
#Stochastic Gradient Descent
pipe = Pipeline([('vect', CountVectorizer()),
                 ('tfidf', TfidfTransformer()),
                 ('model', SGDClassifier())])
SGDmodel = pipe.fit(x_train, y_train)
prediction = SGDmodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
SGDmodel_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

accuracy: 86.23%

7) Gradient Boosting Algorithm:

Gradient boosting is a method standing out for its prediction speed and accuracy, particularly with large and complex datasets. This algorithm has produced the best results. We already know that errors play a major role in any machine learning algorithm. There are mainly two types of error, bias error and variance error. Gradient boost algorithm *helps us minimize bias error* of the model.

Accuracy: 80.71 %

Snippet:

```
#GradientBoostingClassifier
from sklearn.ensemble import GradientBoostingClassifier
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', GradientBoostingClassifier(loss = 'deviance',
                                                         learning_rate = 0.01,
                                                         n_estimators = 10,
                                                         max_depth = 5,
                                                         random_state=55))])

GBCmodel = pipe.fit(x_train, y_train)
prediction = GBCmodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
GBCmodel_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

accuracy: 80.71%

8) XGBClassifier:

[XGBoost](#) is a decision-tree-based ensemble Machine Learning algorithm that uses a [gradient boosting](#) framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks. However, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now.

Accuracy: 80.75 %

Snippet:

```
#####XGBClassifier
from xgboost import XGBClassifier
pipe = Pipeline([('vect', CountVectorizer()),
                 ('tfidf', TfidfTransformer()),
                 ('model', XGBClassifier(loss = 'deviance',
                                         learning_rate = 0.01,
                                         n_estimators = 10,
                                         max_depth = 5,
                                         random_state=2020))])

xgboostmodel = pipe.fit(x_train, y_train)
prediction = xgboostmodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
xgboostmodel_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

```
[13:51:37] WARNING: C:\Users\Administrator\workspace\xgboost-win64_release_1.2.0\src\learner.cc:516:
Parameters: { loss } might not be used.
```

```
accuracy: 80.75%
```

7) Naive Bayes Classifiers:

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other, which assumes that the presence of a particular feature in a class is independent of the presence of any other feature. It provides a way for calculating the posterior probability.

$P(c|x)$ = posterior probability of class given predictor
 $P(c)$ = prior probability of class

$P(x|c)$ = likelihood (probability of predictor given class)

$P(x)$ = prior probability of predictor

i) Multinomial Naive Bayes Classifier:

The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tf-idf may also work.

Accuracy: 78.79 %

Snippet:

```
#####Multinomial Naive Bayes Classifier
pipe = Pipeline([('vect', CountVectorizer()),
                 ('tfidf', TfidfTransformer()),
                 ('model', MultinomialNB())])

MNBCmodel = pipe.fit(x_train, y_train)
prediction = MNBCmodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
Multinomial_Naive_Bayes_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

accuracy: 78.79%

ii) Bernoulli Naive Bayes Classifier:

Bernoulli Naive Bayes is a part of the family of Naive Bayes. It only takes binary values. The most general example is where we check if each value will be whether or not a word appears in a document. That is a very simplified model. In cases where counting the word frequency is less important, Bernoulli may give better results. In simple words, we have to count every value binary term occurrence features i.e. a word occurs in a document or not. These features are used rather than finding the frequency of a word in the document.

Accuracy: 76.08 %

Snippet:

```
#####Bernoulli Naive Bayes Classifier
pipe = Pipeline([('vect', CountVectorizer()),
                 ('tfidf', TfidfTransformer()),
                 ('model', BernoulliNB())])

BNBCmodel = pipe.fit(x_train, y_train)
prediction = BNBCmodel.predict(x_test)
print("accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
Bernoulli_Naive_Bayes_accuracy = round(accuracy_score(y_test, prediction)*100,2)
```

accuracy: 76.08%

7.6 IMPLEMENTATION STEPS

A. Static Search Implementation-

In the static part, we have trained and used 7 algorithms for classification. They are Naïve Bayes Classifier, Random Forest, Logistic Regression, Decision Tree Classifier, Stochastic Gradient Descent, Gradient Boosting Algorithm, XGB Classifier.

Step 1: In the first step, we have extracted features from the already preprocessed dataset. These features are Bag-of-words, Tf-Idf Features and N-grams.

Step 2: Here, we have built all the classifiers for predicting fake news detection. The extracted features are fed into different classifiers. We have used Naive-bayes, Logistic Regression, and Random Forest, XGBClassifier classifiers from sklearn. Each of the extracted features was used in all of the classifiers.

Step 3: Once fitting the model, we compared the score of each classifier and checked the confusion matrix.

Step 4: After fitting all the classifiers, 1 best performing model was selected as candidate models for fake news classification.

Step 5: Finally selected model was used for fake news detection with the probability of truth.

Step 6: Our finally selected and best performing classifier was Logistic Regression which was then saved on disk. It will be used to classify the fake news.

It takes a news article as input from user then model is used for final classification output that is shown to user along with probability of truth.

7.7 EVALUATION MATRICES

Evaluate the performance of algorithms for fake news detection problems; various evaluation metrics have been used. In this subsection, We review the most widely used metrics for fake news detection. Most existing approaches consider the fake news problem as a classification problem that predicts whether a news article is fake or not:

True Positive (TP): when predicted fake news pieces are classified as fake news.

True Negative (TN): when predicted true news pieces are classified as true news.

False Negative (FN): when predicted true news pieces are classified as fake news.

False Positive (FP): when predicted fake news pieces are classified as true news.

A. Confusion Matrix:

A 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. It allows the visualization of the performance of an algorithm. A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which your classification The model is confused when it makes predictions. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

Table 1: Confusion Matrix

TOTAL	CLASS 1	CLASS 2
CLASS 1 (actual)	TP	FN
CLASS 2 (actual)	FP	TN

By formulating this as a classification problem, we can define following metrics-

1. Precision

2. Recall

3. Accuracy

These metrics are commonly used in the machine learning community and enable us to evaluate the performance of a classifier from different perspectives. Specifically, accuracy measures the similarity between predicted fake news and real fake news.

A. RESULTS

Implementation was done using the above algorithms with Vector features- Count Vectors and Tf-Idf vectors at Word level and Ngram-

level. Accuracy was noted for all models. We used K-fold cross validation technique to improve the effectiveness of the models.

B. Dataset split using K-fold cross validation

This cross-validation technique was used for splitting the dataset randomly into k-folds. (k-1) folds were used for building the model while kth fold was used to check the effectiveness of the model. This was repeated until each of the k-folds served as the test set. I used 3- fold cross validation for this experiment where 67% of the data is used for training the model and remaining 33% for testing.

B. Confusion Matrices for Static System After applying various extracted features (Bag-of-words, Tf-Idf, N-grams) on three different classifiers (Naïve bayes, Logistic Regression and Random Forest), their confusion matrix showing actual set and predicted sets are mentioned below:

Table 2: Confusion Matrix for Naive Bayes Classifier using TF-IDF features

TOTAL = 10240	FAKE (predicted)	TRUE (predicted)
FAKE (actual)	841	3647
TRUE (actual)	427	5325

Table 3: Confusion Matrix for Logistic Regression using TF-IDF features

Total= 10240	Fake (Predicted)	True (Predicted)
Fake (Actual)	1617	2871
True (Actual)	1097	4655

Table 4: Confusion Matrix for Random Forest Classifier using Tf-Idf features-

Total= 10240	Fake (Predicted)	True (Predicted)
Fake (Actual)	1979	2509
True (Actual)	1630	4122

Table 5: Comparison of Precision, Recall, F1-scores and Accuracy for all three classifiers-

Classifiers	Precision	Recall	F1-Score	Accuracy
Naïve Bayes	0.59	0.92	0.72	0.60
Random Forest	0.62	0.71	0.67	0.59
Logistic Regression	0.69	0.83	0.75	0.65

As evident above our best model came out to be Logistic Regression with an accuracy of 65%. Hence we then used grid search parameter optimization to increase the performance of logistic regression which then gave us the accuracy of 80%. Hence we can say that if a user feeds a particular news article or its headline in our model, there are 80% chances that it will be classified to its true nature.

C. Confusion Matrix for Dynamic System

We used real_or_fake.csv with passive aggressive classifier and obtained the following confusion matrix-

Table 6: Confusion Matrix for passive aggressive Classifier-

Total= 1267	Fake (Predicted)	True (Predicted)
Fake (Actual)	588	50
True (Actual)	42	587

Table 7: Performance measures-

Classifier	Precision	Recall	F1-Score	Accuracy
PAC	0.93	0.9216	0.9257	0.9273

8. FUTURE SCOPE

Social media for news consumption is a double-edged sword. On the one hand, its low cost, easy access, and rapid dissemination of information lead people to seek out and consume news from social media. On the other hand, it enables the wide spread of "fake news", i.e., low quality news with intentionally false information. The extensive spread of fake news has the potential for extremely negative impacts on individuals and society

News Spread because they are attractive, whether they are good or bad, right, or wrong, we're living in the age of fake news. Fake news consists of deliberate misinformation under the guise of being authentic news, spread via some communication channel, and produced with a particular objective like generating revenue, promoting, or discrediting a public figure, a political movement, an organization, etc.

Massive amounts of data gave birth to AI systems that are already producing human-like synthetic texts, powering a new scale of disinformation operation. Based on Natural Language Processing (NLP) techniques, several lifelike text-generating systems have proliferated, and they are becoming smarter every day.

Technology will help in this fight. AI makes it possible to find words and patterns that indicate fake news in huge volumes of data, and tech companies are already working on it.

We would further develop this system by working on the below modules and features

Use web scraping to scrape real time social media posts and news to curate a real time dataset for the system. This data set can be used to Curate new and a variety of data to train this can improve the accuracy of the system.

Create a community driven news blog site to determine the authenticity and trueness of the news published. An admin authenticated blog posts which curates only the true news articles for the end users.

Create extension apps to integrate with social media applications to perform real time news prediction and analysis for the posts and news articles posted over the platform to determine the trueness of the message/posts.

Algorithms such as fake news detections can be adopted by the large social media platforms to integrate itself into a platform based on only true events where the false news circulated can be detected and determined to the users in real time.

9. CONCLUSION

Fake news detection has many open issues that require the attention of researchers. For instance, in order to reduce the spread of fake news, identifying key elements involved in the spread of news is an important step. Graph theory and machine learning techniques can be employed to identify the key sources involved in the spread of fake news. Likewise, real time fake news identification in videos can be another possible future direction.

This Model is Successfully able to classify fake and real news with 0 and 1 label respectively. The accuracy of the model with Multinomial Naive Bayes algorithm was found 94.5% and with Passive Aggressive the accuracy was further enhanced and clocked 99.5%.

Though the Spread of fake news through different channels will not steer down, preventive measures can be taken to avoid oneself from being misguided. Fake news will persist if there is no governing body to verify the news from its source. The truthfulness can be measured on various grounds as to which crowd is a renowned newspaper organization.

In the Study it was found that most of the fake news present in the datasets were political. Therefore, It can be concluded that there are chances of coming across a fake news article which is related to politics.

Efficiency of the model can further be improved by training the model with a dataset holding the latest news.

10. REFERENCES

Datasets: True.csv, Fake.csv

1. International journal of recent technology and engineering (IJRTE) ISSN: 2277-3878, volume-7, issue-6, March 2019
2. Shloka Gilda, “Evaluating Machine Learning Algorithms for Fake News Detection”
3. 2017 IEEE 15th Student Conference on Research and Development (SCOREd).
4. Mykhailo Granik, Volodymyr Mesyura, “Fake News Detection Using Naive Bayes Classifier”, 2017 IEEEFirst Ukraine Conference on Electrical and Computer Engineering (UKRCON).
5. Gravanis, G., et al., Behind the cues: A benchmarking study for fake news detection. Expert Systems with Applications, 2019. 128: p. 201- 213.
6. Zhang, C., et al., Detecting fake news for reducing misinformation risks using analytics approaches. European Journal of Operational Research, 2019.
7. Bondielli, A. and F. Marcelloni, A survey on fake news and rumor detection techniques. Information Sciences, 2019. 497: p. 38-55
8. Ko, H., et al., Human-machine interaction: A case study on fake news detection using backtracking based on a cognitive system. Cognitive Systems Research, 2019. 55: p. 77- 81.
9. Zhang, X. and A.A. Ghorbani, An overview of online fake news: Characterization, detection, and discussion. Information Processing & Management, 2019.
10. Robbins, K.R., W. Zhang, and J.K. Bertrand, The ant colony algorithm for feature selection in high-dimension gene expression data for disease classification. Journal of Mathematical Medicine and Biology, 2008

11. Alirezaei, M., S.T.A. Niaki, and S.A.A. Niaki, A bi-objective hybrid optimization algorithm to reduce noise and data dimension in diabetes diagnosis using support vector machines. *Expert Systems with Applications*, 2019. 127: p. 47-57
12. Zakeri, A. and A. Hokmabadi, Efficient feature selection method using real-valued grasshopper optimization algorithm. *Expert Systems with Applications*, 2019. 119: p. 61-7