FAKE NEWS DETECTION

Submitted by:

Ankita Ramdas Mhetre

ACKNOWLEDGMENT

I would like to express my deep and sincere gratitude towards Fliprobo technologies for providing me the internship opportunity and a great chance for learning and professional development.

I express my deepest gratitude and special thanks to the subject matter expert (SME), Mr Shubham Yadav who in spite of being extraordinarily busy with his duties, took time out to hear, guide and keep me on correct path ,motivated me for taking part in useful decision & giving necessary advices to do the project and providing invaluable guidance throughout.

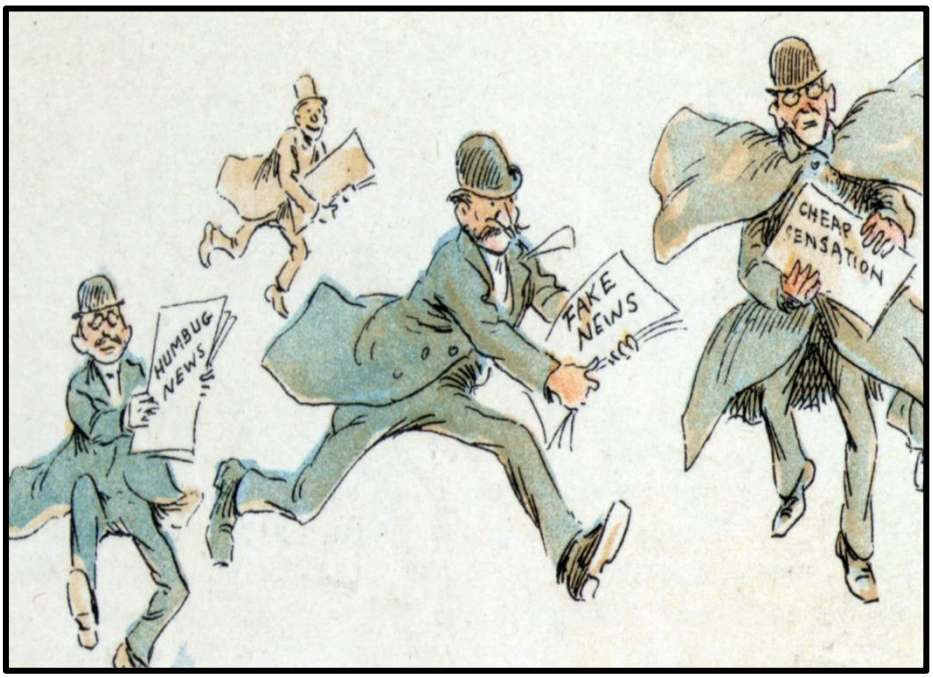
His dynamism, vision, sincerity and motivation have deeply inspired me. He has taught me the methodology to carry out the task and to present the project works as clearly as possible. It was a great privilege and honour to work and study under his guidance. I am extremely grateful for what he has offered me.

He was never too busy to spare his valuable time for this work. No words are adequate to express my gratitude towards him. I would also like to thank him for his friendship and empathy.

INTRODUCTION

• Business Problem Framing

The authenticity of Information has become a longstanding issue affecting businesses and society, both for printed and digital media. On social networks, the reach and effects of information spread occur at such a fast pace and so amplified that distorted, inaccurate, or false information acquires a tremendous potential to cause real-world impacts, within minutes, for millions of users. Recently, several public concerns about this problem and some approaches to mitigate the problem were expressed.



In this project, we are given a dataset to build a model to predict whether news is fake or not fake. Our sole objective is to classify the news from the dataset as fake or true news. Perform extensive EDA of news and selecting and building a powerful model for classification.

• Conceptual Background of the Domain Problem

Fake news refers to misinformation or disinformation in the country which is spread through word of mouth and traditional media and more recently through digital forms of communication such as edited videos, memes, unverified advertisements and social media propagated rumours. Fake news spread through social media in the country has become a serious problem, with the potential of it resulting in mob violence, as was the case where at least 20 people were killed in 2018 as a result of misinformation circulated on social media.

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

News media has become a channel to pass on the information of what's happening in the world to the people living. Often people perceive whatever conveyed in the news to be true. There were circumstances where even the news channels acknowledged that their news is not true as they wrote. But some news has a significant impact not only on the people or government but also on the economy. One news can shift the curves up and down depending on the emotions of people and political situation.

Rasmus Kleis Nielsen, director at Reuters Institute for the Study of Journalism, thinks that "the problems of disinformation in a society like India might be more sophisticated and more challenging than they are in the West". The damage caused due to fake news on social media has increased due to the growth of the internet penetration in India, which has risen from 137 million internet users in 2012 to over 600 million in 2019.India is the largest market for WhatsApp, with over 230 million users, and as a result it is one of the main platforms on which fake news is spread. One of the principal problems is that receivers believe anything sent to them over social media due to lack of awareness. Various initiatives and practices have been started and adopted to curb the spread and impact of fake news. Fake news is also spread through Facebook and Twitter.

According to a report by The Guardian, the Indian media research agency CMS stated that the cause of spread of fake news was that India "lacked media policy for verification". Additionally, law enforcement officers have arrested reporters and journalists for "creating fictitious articles", especially when the articles were controversial.

In India, the spread of fake news has occurred with relation to political and religious matters. For an example the IT Cells of the BJP, Congress and other political parties have been accused of spreading fake news against the party's political opponents and any campaigns against the party. The BJP was accused of spreading fake news targeting religious minorities. RSS mouthpiece Organizer and Congress mouthpiece National Herald have also been accused of misleading reports.

It is important to identify the fake news from the real true news. The problem has been taken over and resolved with the help of Natural Language Processing tools which help us identify fake or true news based on historical data. The news is now in safe hands!

- ## Review of Literature

Societal issues are being raised about the individuals' ability to tell apart what is fake and what is authentic, while surfing and actively engaging in information overloaded networks.

As reported by Anderson, youngsters are known to be tech-savvy when compared to their parents, but when it comes to the ability to tell if a news piece is fake or not, they seem as confused as the rest of the society and 44% have confirmed it in a research conducted by Common Sense Media

Thirty-one per cent of kids aged 10 to 18 shared a news story online that they later found out were wrong or inaccurate, while only 44% say they can tell a fake news story from a real one, research published by Common Sense Media finds. This situation raises a whole new dimension of concerns related to digital literacy that go beyond the mere ability to access and manage technology.

Together with societal challenges, there is a dramatic and inconspicuous situation happening in the media landscape, the public sphere and journalism industry that requires debate and examination, pointing out two main aspects. The first one relies on the fact that news publishers have lost control over the distribution of news, which are presented to Internet users by obscure and unpredictable algorithms. Also, news market newcomers (such as Buzz Feed, Vox and Fusion) have built their presence by embracing these technologies, undermining the long-term positions occupied by more traditional news publishers. The second aspect relies on the increasing power that social media companies, such as Google, Apple, Facebook and Amazon, have gained in controlling who publishes what to whom, and how the publications are monetized.

The fake news subject has become so prevalent that the Commons Culture, Media and Sport Committee is currently investigating concerns about the public being swayed by propaganda and untruths. The journalism is also at stake, since an increasing proportion of the adults are getting their news from social media and fictional stories are presented in such way that it can be very difficult to tell them apart from what is authentic.

The FiB system: During a 'hackathon' at Princeton University, four college students created a browser extension to detect fake news in 36 hours. They named their project "FiB: Stop living a lie". The system checks the personal news feed and labels each post according to the system's checked authenticity (verified or non-verified). Basically, the algorithm works based on the following process:

1) For the text as post content, it is used as a search query on Google/Bing. The retrieved searches with high confidence are then summarized and shown to the user.

2) For links, it considers the website's reputation, also querying it against malware and phishing websites databases.

3) For pictures like Twitter snapshots, it first converts the image to text; then it uses the usernames mentioned in the tweet, to get all tweets of these users, and check if current tweet was ever posted by any of the users. The browser plug-in then adds a little tag in the top-right corner that says whether the story is verified, or not (as illustrated in Figures below).

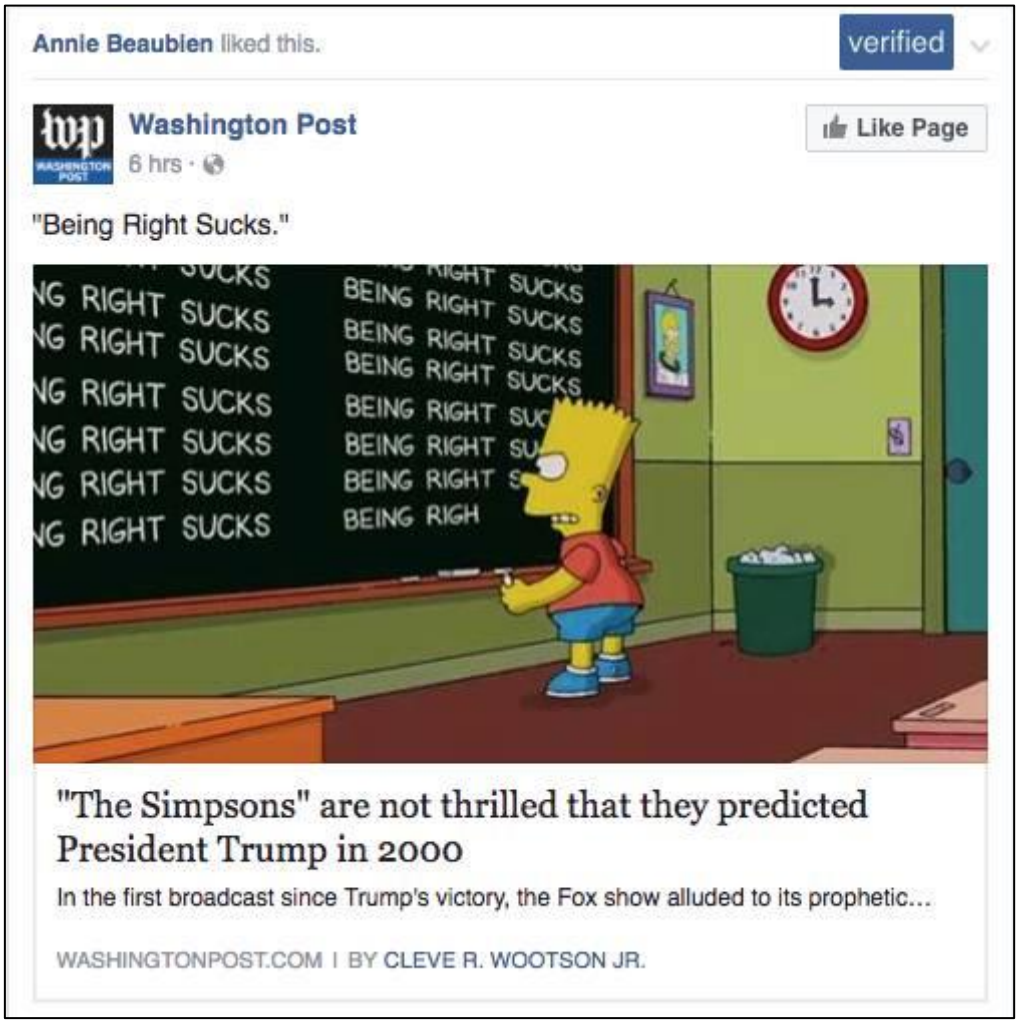


Fig: Verified post

Fig: not verified post

The Facebook Fake News Task Team and their approach in November 2016 it was reported that Facebook had formed an unofficial task force working on the problem of fake news. Through their official channels, Facebook promised to reprioritize fake news on its pages, saying one of their news feed values is “authentic communication” and that it's acting to prevent posts that are “misleading, sensational or spam”. Pages that have been posting fake news have been studied by Facebook's experts and are now expected to be seen less frequently in news feeds.

Facebook has faced an increasing criticism over its role in the 2016 US presidential election because it allowed the propagation of fake news disguised as news stories coming from unchecked websites. This spreading of false information during the election cycle was so severe that Facebook was labelled as "dust cloud of nonsense." Refer the following article for detail reading.

(it-only-took-36-hours-for-these-students-to-solve- facebooks-fake-news-problem).

The fact is that the presidential election year has shown how the lines have blurred between facts and speculation, with people profiting off the spread of fake news. There were more than 100 news sites that made up pro-Trump content traced to Macedonia, according to a BuzzFeed News investigation.

Counter Measures

Internet shutdowns are used by the government as a way to control social media rumours from spreading. Ideas such as linking Aadhaar to social media accounts has been suggested to the Supreme Court of India by the Attorney General. In some parts of India like Kannur in Kerala, the government conducted fake news classes in government schools. Some say the government should conduct more public-education initiatives to make the population more aware of fake news.

In India, Facebook has partnered with fact-checking websites such as BOOM[and Webqoof by The Quint. Following over 30 killings linked to rumours spread over WhatsApp, WhatsApp introduced various measures to curb the spread of misinformation, which included limiting the number of people a message could be forwarded to as well as introducing a tip-line among other measures such as suspending accounts and sending cease-and-desist letters. WhatsApp also added a small tag, forwarded, to relevant messages. They also started a course for digital literacy and came out with full page advertisements in newspapers in multiple languages. Twitter has also taken action to curb the spread of fake news such as deleting accounts.

In 2018, Google News launched a program to train 8000 journalists in seven official Indian languages including English. The program, Google's largest training initiative in the world, would spread awareness of fake news and anti-misinformation practices such as fact-checking.

Fact-checking in India has become a business, spurning the creation of fact-checking websites such as BOOM, Alt News, Factly and SMHoaxSlayer.

Media houses also have their own fact-checking departments now such as the India Today Group, Times Internet has TOI Factcheck and The Quint has WebQoof.

In November 2019, the Indian ministry of information and broadcasting planned to set up a FACT checking module to counter the circulation of fake news by continuous monitoring of online news sources and publicly visible social media posts. The module will work on the four principles of "Find, Assess, Create and Target" (FACT). The module will initially will be run by information service officers. Near the end of 2019, the Press Information Bureau (which comes under the Ministry of Information and Broadcasting) set up a fact-checking unit which would focus on verifying news related to the government.

- ## Motivation for the Problem Undertaken

In the news industry, in particular, but also in society at large, fake news detection has become a central discussion topic, as the need to permanently assess the veracity of digital content has been raised by the constant spread of false news / information. Information veracity is a long-term issue affecting society both for printed and digital media. The sensationalism of not-so-accurate eye catching and intriguing headlines aimed at retaining the attention of audiences to sell information has persisted all throughout the history of all kinds of information broadcast. On social networking websites, the reach and effects of information spread are however significantly amplified and occur at such a fast pace, that distorted, inaccurate or false information acquires a tremendous potential to cause real impacts, within minutes, for millions of users.

In this modern world, data is very important and by the 2020 year, 1.7 megaBytes data generated per second. So there are many technologies that change the world by this large amount of data. Machine learning is one of them and we are using this technology to detect fake news.

Internet is one of the important inventions and a large number of persons are its users. These persons use this for different purposes. There are different social media platforms that are accessible to these users. Any user can make a post or spread the news through these online platforms. These platforms do not verify the users or their posts. So some of the users try to spread fake news through these platforms. These fake news can be a propaganda against an individual, society, organization or political party. A human being is unable to detect all these fake news. So there is a need for machine learning classifiers that can detect these fake news automatically.

This project work makes an analysis of the research related to fake news detection and explores the traditional machine learning models to choose the best, in order to create a model of a product with supervised machine learning algorithm, that can classify fake news as true or false, by using tools like python scikit-learn, NLP for textual analysis.

Fake news just not only affects an individual but it can also affect an organization or business. So controlling the fake news is mandatory. A person can know the news is fake only when he knows the complete story of that topic. It is a difficult task because most of the people do not know about the complete story and they just start believing in the fake news without any verification. The question arises here how to control fake news because a person cannot control the fake news. The answer is machine learning. Through the use of machine learning these fake news can be detected easily and automatically .Once someone will post the fake news, machine learning algorithms will check the contents of the post and will detect it as fake news.

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 features of the language and content only within the source in question, without utilizing any fact checker or knowledge base

Analytical Problem Framing

- ## Mathematical/ Analytical Modelling of the Problem

As the dataset was completely textual we did not had much scope of performing statistical analysis on the data .The only exploratory analysis were carried out in order to get the overview of the data. The few basic steps could be seen in below snapshots

```
#checking the size of the data
df.shape
```

```
(20800, 6)
```

```
# chceking the columns present in dataset
df.columns
```

```
Index(['Unnamed: 0', 'id', 'headline', 'written_by', 'news', 'label'], dtype='object')
```

```
#checking the datatype
df.dtypes
```

```
Unnamed: 0     int64
id             int64
headline      object
written_by    object
news          object
label          int64
dtype: object
```

```
df.dtypes.value_counts()
```

```
int64     3
object    3
dtype: int64
```

```
#checking for null values
df.isnull().sum()
```

```
Unnamed: 0       0
id               0
headline       558
written_by    1957
news            39
label            0
dtype: int64
```

```
#drop any duplicate values from the dataframe
#print("original shape=",df.shape)
df.drop_duplicates(subset=None,inplace=False)
df.head()
print("after removing duplicates shape=",df.shape)
```

- **Data Sources and their formats**

There are 6 columns in the dataset provided. The description of each of the column is given below:

1) **id:** Unique id of each news article
2) **headline**: It is the title of the news.
3) **news:** It contains the full text of the news article
4) **Unnamed:0**: It is a serial number
5) **written by:** It represents the author of the news article
6) **label**: It tells whether the news is fake (1) or not fake (0).

	Unnamed: 0	id	headline	written_by	news	label
0	0	9653	Ethics Questions Dogged Agriculture Nominee as...	Eric Lipton and Steve Eder	WASHINGTON — In Sonny Perdue's telling, Geo...	0
1	1	10041	U.S. Must Dig Deep to Stop Argentina's Lionel ...	David Waldstein	HOUSTON — Venezuela had a plan. It was a ta...	0
2	2	19113	Cotton to House: 'Do Not Walk the Plank and Vo...	Pam Key	Sunday on ABC's "This Week," while discussing ...	0
3	3	6868	Paul LePage, Besieged Maine Governor, Sends Co...	Jess Bidgood	AUGUSTA, Me. — The beleaguered Republican g...	0
4	4	7596	A Digital 9/11 If Trump Wins	Finian Cunningham	Finian Cunningham has written extensively on...	1
5	5	3196	Whatever the Outcome on November 8th the US Wi...	NaN	Taming the corporate media beast Whatever the ...	1
6	6	5134	Rapid Evolution Saved This Fish From Pollution...	JoAnna Klein	The State of New Jersey says you can't eat the...	0

Fig: Sample instance of the dataset

The data types of the columns were as follows:

```
#checking the datatype
df.dtypes
```

```
Unnamed: 0     int64
id             int64
headline      object
written_by    object
news          object
label          int64
dtype: object
```

```
df.dtypes.value_counts()
```

```
int64     3
object    3
dtype: int64
```

The dimension of data was 20800, 6 i.e. it had 20800 rows and 6 columns.

- ## Data Pre-processing Done

In any Machine learning task, cleaning or pre-processing the data is as important as model building. Text data is one of the most unstructured forms of available data and when comes to deal with Human language then it's too complex. Have you ever wondered how Alexa, Siri, Google assistant can understand, process, and respond in Human language? NLP is a technology that works behind it where before any response lots of text pre-processing takes place.

Text pre-processing is a method to clean the text data and make it ready to feed data to the model. Text data contains noise in various forms like emotions, punctuation, and text in a different case. When we talk about Human Language then, there are different ways to say the same thing, and this is only the main problem we have to deal with because machines will not understand words, they need numbers so we need to convert text to numbers in an efficient manner.

The various techniques used in our dataset for cleaning are as follows:

1) **Lowercasing**

Text often has a variety of capitalization reflecting the beginning of sentences or proper nouns emphasis. The common approach is to reduce everything to lower case for simplicity. For example, words like Ball and ball are treated differently by machine. So, we need to make the text in the same case and the most preferred case is a lower case to avoid such problems.

```
#lowercase all the text columns
df["headline"]=df['headline'].str.lower()
df['news']=df['news'].str.lower()
df["written_by"]=df["written_by"].str.lower()
```

Lowercasing is applicable to most text mining and NLP tasks and significantly helps with consistency of the output. However, it is important to remember that some words, like “US” and “us”, can change meanings when reduced to the lower case.

`

2) Noise Removal

Noise removal refers to removing characters digits and pieces of text that can interfere with the text analysis. There are various ways to remove noise, including punctuation removal, special character removal, and numbers removal, html formatting removal, domain specific keyword removal, source code removal, and more.

Noise removal is highly domain dependent. For example, in tweets, noise could be all special characters except hash tags as they signify concepts that can characterize a tweet. We should also remember that strategies may vary depending on the specific task: for example, numbers can be either removed or converted to textual representations.

Here are few techniques used for noise removal

```
#removing html tags
import re
def remove_html(headline_text):
    html=re.compile(r'<.*?>')
    return html.sub(r'',headline_text)
```

```
#removing punctuations for text
def remove_punctuations(text):
    return re.sub(r'[^\w\s]','',text)
```

```
#keep only characters from Aa to Zz and numbers
def cleaning(text):
    return re.sub('[^A-Za-z0-9]+', ' ',text)
```

3) Stop-word removal

Stop words are a set of commonly used words in a language like “a”, “the”, “is”, “are” and etc. in English. These words do not carry important meaning and are removed from texts in many data science tasks. The intuition behind this approach is

that, by removing low information words from text, we can focus on the important words instead. Besides, it reduces the number of features in consideration which helps keep your models better sized.

```
#import library for stopwords removal
from nltk.corpus import stopwords

#removing stopwords
stop_words = set(stopwords.words('english'))
re_stop_words = re.compile(r"\b(" + "|".join(stop_words) + ")\\W", re.I)
def removeStopWords(sentence):
    global re_stop_words
    return re_stop_words.sub(" ", sentence)
#apply the remove_stopwords function
df['headline'] = df['headline'].apply(removeStopWords)
df['news']=df['news'].apply(removeStopWords)
df["written_by"]=df["written_by"].apply(removeStopWords)
df.head()
```

Stop word removal is commonly applied in search systems, text classification applications, topic modelling, topic extraction and others. Stop word lists can come from pre-established sets or we can create a custom one for our domain.

4) Stemming and lemmatization

Stemming is the process of eliminating affixes (suffixes, prefixes, infixes, circumfixes) from a word in order to obtain a word stem. The results can be used to identify relationships and commonalities across large datasets. There are several stemming models, including Porter and Snowball. The danger here lies in the possibility of over stemming where words like “universe” and “university” are reduced to the same root of “univers”.

Lemmatization

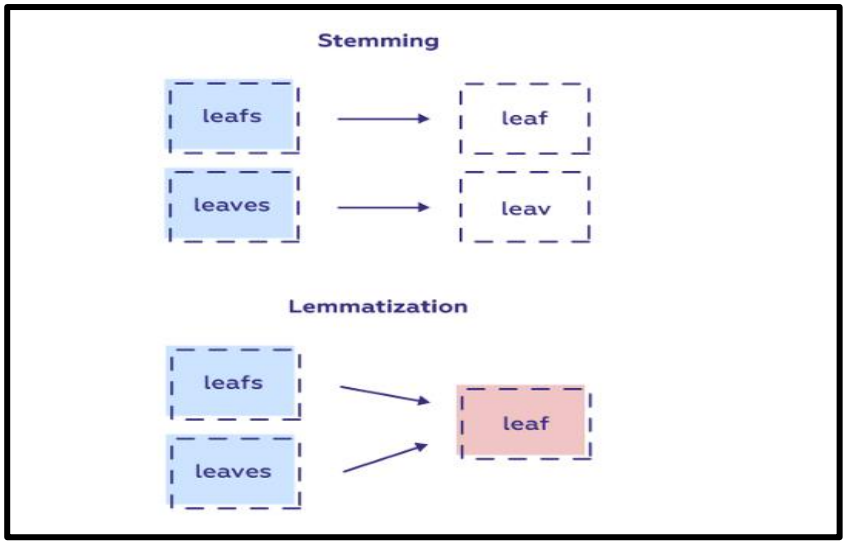
Lemmatization is related to stemming, but it is able to capture canonical forms based on a word’s lemma. By determining the part of speech and utilizing special tools, like WordNet’s lexical database of English, lemmatization can get better results:

The stemmed form of leafs is: leaf

The stemmed form of leaves is: leav

The lemmatized form of leafs is: leaf

The lemmatized form of leaves is: leaf



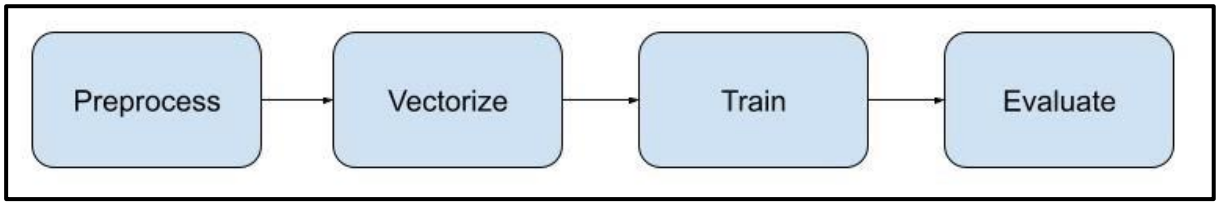
Stemming may be more useful in queries for databases whereas lemmatization may work much better when trying to determine text sentiment.

Lemmatization is similar to stemming, used to stem the words into root word but differs in working. Actually, Lemmatization is a systematic way to reduce the words into their lemma by matching them with a language dictionary.

```
#Lemmatization
from nltk.stem import WordNetLemmatizer
def lemma_txt(text):
    ss = WordNetLemmatizer()
    return "".join([ss.lemmatize(w) for w in text])

#apply lemma_txt
df['headline'] = df['headline'].apply(lemma_txt)
df['news']=df['news'].apply(lemma_txt)
df["written_by"]=df["written_by"].apply(lemma_txt)
df.head()
```

5) Word Vectorization

Word Embedding or Word vectorization is a methodology in NLP to map words or phrases from vocabulary to a corresponding vector of real numbers which used to find word predictions, word similarities/semantics. The process of converting words into numbers is called Vectorization.



1. Count Vectorizer

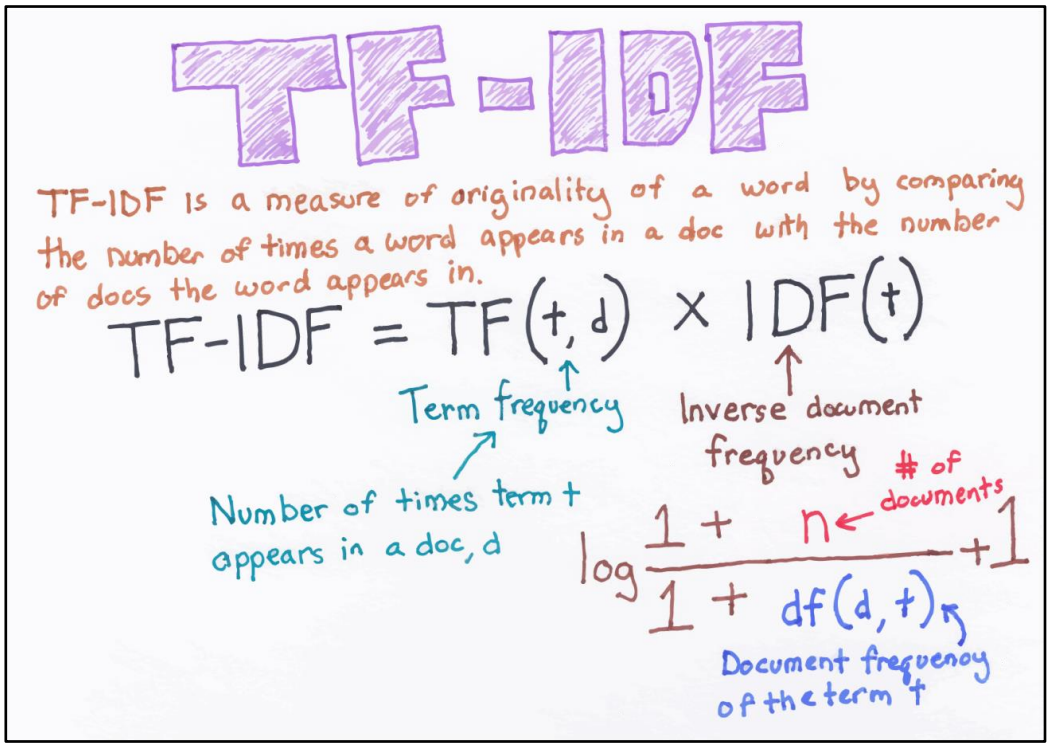
CountVectorizer is used to convert a collection of text documents to a vector of term/token counts. It also enables the pre-processing of text data prior to generating

the vector representation. This functionality makes it a highly flexible feature representation module for text.

	the	red	dog	cat	eats	food
1. the red dog →	1	1	1	0	0	0
2. cat eats dog →	0	0	1	1	1	0
3. dog eats food →	0	0	1	0	1	1
4. red cat eats →	0	1	0	1	1	0

2. Tf-IDF Vectorizer

TF-IDF is an abbreviation for Term Frequency Inverse Document Frequency. This is very common algorithm to transform text into a meaningful representation of numbers which is used to fit machine algorithm for prediction.



Tf-idf similar to CountVectorizer but higher weights are given to uncommon words, as they have more influence over classification.

```
#transforming words to vectors
from sklearn.feature_extraction.text import TfidfVectorizer
tf = TfidfVectorizer(max_features =2000)
x = tf.fit_transform(x)
x
```

```
<20761x2000 sparse matrix of type '<class 'numpy.float64'>'
        with 3197326 stored elements in Compressed Sparse Row format>
```

```
# visualize the sparse matrix we obtained after tf-idf transform (feature names)
df_features=pd.DataFrame(x.toarray(),columns=tf.get_feature_names())
df_features
```

10	100	11	12	13	14	15	16	17	...	yes	yet	york
0.000000	0.084305	0.000000	0.0	0.024600	0.000000	0.000000	0.00000	0.0	...	0.0	0.000000	0.012325
0.028846	0.000000	0.069951	0.0	0.000000	0.000000	0.067839	0.00000	0.0	...	0.0	0.000000	0.019848
0.000000	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.00000	0.0	...	0.0	0.000000	0.000000
0.000000	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.00000	0.0	...	0.0	0.024577	0.018027
0.000000	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.00000	0.0	...	0.0	0.000000	0.000000
...	...	...	...	...	...	...	...	...	...	...	...	...
0.000000	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.00000	0.0	...	0.0	0.000000	0.000000
0.000000	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.00000	0.0	...	0.0	0.000000	0.000000
0.000000	0.000000	0.000000	0.0	0.030975	0.030948	0.000000	0.03103	0.0	...	0.0	0.000000	0.000000

- ## Hardware and Software Requirements and Tools Used

Open source web-application used for programming:

1. **Jupyter Notebook**

   In programming, a module is a piece of software that has a specific functionality. A module can contain executable statements as well as function definitions. These statements are intended to initialize the module. They are executed only the first time the module name is encountered in an import statement. Modules can import other modules. Packages are a way of structuring Python's module namespace by using "dotted module names". For example, the module name A.B designates a sub module named B in a package named A

Python Libraries / Packages used were:

1. **Pandas**: pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

   We have used pandas to import the csv file using pd.read_csv all data analysis have been done using the pandas and numpy libraries. The data characteristics have been studied using pandas functions like pd.shape(), pd.dtypes, pd.columns etc.

2. **NumPy:** NumPy is an open-source numerical Python library. NumPy contains a multi-dimensional array and matrix data structures. It can be utilised to perform a number of mathematical operations on arrays such as trigonometric, statistical, and algebraic.

3. **Matplotlib**: library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

   The Matplotlib libraries pyplot function is used for making plots ,plt.show() that has been used is a part of matplotlib library.

4. **Seaborn**: Seaborn is a Python data visualization library based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

   All the visualizations made are built using the saeborn library. Alias used for seaborn is sns.

5. **NLTK**: NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum. NLTK has been called "a wonderful tool for teaching, and working in, computational linguistics using Python," and "an amazing library to play with natural language."

6. **Re**: regular expression library, a regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern.
   Regular expressions are widely used in UNIX world. The Python
   module re provides full support for Perl-like regular expressions in Python.

7. **Wordcloud**: Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analysing data from social network websites.

8. **Sklearn:** Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python.
   It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.
   All the Machine Learning model have been imported from the sklearn package
   The evaluation metrics like classification reports and accuracy score etc., functions are also imported from the same.

Model/s Development and Evaluation

- **Identification of possible problem-solving approaches (methods)**

1) We began with exploring the dataset; our aim was to build a model which could classify the news as fake or real on the basis of news and the headline of news.
2) The dataset had 20800 rows and 6 columns named id, unnamed 0, headline, news, written_by and label. The label was our dependent variable.

	Unnamed: 0	id	headline	written_by	news	label
0	0	9653	Ethics Questions Dogged Agriculture Nominee as...	Eric Lipton and Steve Eder	WASHINGTON — In Sonny Perdue's telling, Geo...	0
1	1	10041	U.S. Must Dig Deep to Stop Argentina's Lionel ...	David Waldstein	HOUSTON — Venezuela had a plan. It was a ta...	0
2	2	19113	Cotton to House: 'Do Not Walk the Plank and Vo...	Pam Key	Sunday on ABC's "This Week," while discussing ...	0
3	3	6868	Paul LePage, Besieged Maine Governor, Sends Co...	Jess Bidgood	AUGUSTA, Me. — The beleaguered Republican g...	0
4	4	7596	A Digital 9/11 If Trump Wins	Finian Cunningham	Finian Cunningham has written extensively on...	1
5	5	3196	Whatever the Outcome on November 8th the US Wi...	NaN	Taming the corporate media beast Whatever the ...	1
6	6	5134	Rapid Evolution Saved This Fish From Pollution...	JoAnna Klein	The State of New Jersey says you can't eat the...	0

3) There were 3 integer columns and 3 object columns in the data
4) Upon checking for the null values we got the below output:

```
#checking for null values
df.isnull().sum()
```

```
Unnamed: 0        0
id                0
headline        558
written_by     1957
news             39
label             0
dtype: int64
```

5) The code snippet says there were 558 null values in "headline"column,1957 in "written_by" and 39 Nan values in "news
6) In order to handle these null values, we decided to drop all 39 nulls in "news" column because there is no use of empty news.
7) For rest 2 columns i.e. "headline" and "written by" we have substituted a separate text as "text unavailable"

```
# drop the null values in news
# it had only 39 null values
df.dropna(subset=['news'],inplace=True)
```

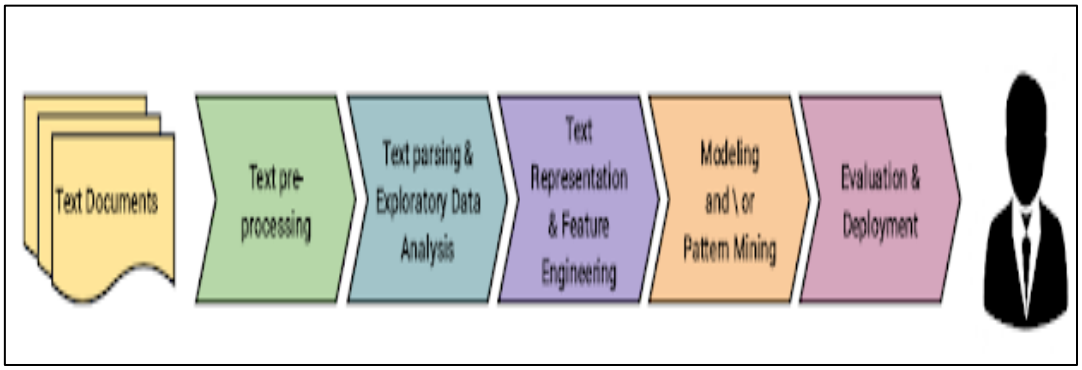
```
#for rest two columns i.e headlines and written by we will add a seperate field there as "Text unavailable"
df=df.fillna("text unavailable")
```

```
#reset the index after dropping the null value
df.reset_index(inplace=True)
```

8) The unnamed 0 and id column had unique value for each row hence we decided to drop them as they were of no use.

```
#drop columns "unnamed 0" and "id"
df.drop(columns=["Unnamed: 0","id"],axis=1,inplace=True)
df.sample(5)
```

9) The next very important step was to carry out the pre-processing on text data in order to transform the raw data into meaningful text and extract useful and relevant information from the text available.



10) Various user defined functions were created to clean up text: remove_punctautions, cleaning, remove_emojis, remove_html etc.
11) The later part included stop words removal and lemmatization.
12) Once we were ready with clean text data, the next step was to transform the entire textual data into some vectors as our machines could only interpret numbers.
13) As we all know that transformers such as sklearn.preprocessing. OneHotEncoder and sklearn.preprocessing.StandardScaler can operate on multiple data columns simultaneously.
14) sklearn.feature_extraction.text.TfidfVectorizer on the other hand, can only process one column at a time, so you need to make a new transformer for each text column in your dataset. This can get a little tedious and in particular makes pipelines more verbose.
15) To make the task easier we have concatenated our 3 text columns into a single column so as to pass it to tf-idf and get the vectors. It seemed a good idea. An instance is shown below:

```
#combining all three columns together and assign the combined corpus to x(independent variable)
x= (df['written_by'] + ' ' + df['headline'] + ' ' +df['news'])
```

```
x.head()
```

```
0    eric lipton  steve eder ethics questions dogge...
1    david waldstein us must dig deep  stop argenti...
2    pam key cotton  house   walk  plank  vote   bi...
3    jess bidgood paul lepage besieged maine govern...
4    finian cunningham  digital 911  trump wins  fi...
dtype: object
```

16) The tfidf was then performed on “X” and “label” column was passed to y variable
17) Then we made a train –test split with 805 of data for training and remaining 20% was the test data. A random state of 42 was randomly chosen with stratify for label.

```
#splitting data into training and testing
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42,stratify=y)
print("Train shapes : X = {}, y = {}".format(x_train.shape,y_train.shape))
print("Test shapes : X = {}, y = {}".format(x_test.shape,y_test.shape))
```

```
Train shapes : X = (16608, 2000), y = (16608,)
Test shapes : X = (4153, 2000), y = (4153,)
```

18) The final step was importing necessary model building libraries and implementing various algorithms and save the best working model.
19) The detail working and metrics are studied further in this report.

- ## Testing of Identified Approaches (Algorithms)

The algorithms used for the training and testing were:

1) Logistic regression
2) Passive aggressive classifier
3) KNeighbors classifier
4) MultinomialNB
5) Support vector classifier(SVC)
6) GradientBoosting classifier

All the models were trained with hyper-parameters obtained by GridSearch Cv. The libraries imported were as follows

```
#importing libraries for model building
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier

#boosting model

from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import GridSearchCV

# metrics for evaluation
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
from sklearn.model_selection import cross_val_score
from sklearn .metrics import roc_auc_score,auc,roc_curve
```

- **Run and Evaluate selected models**

1) Logistic Regression

```
#hypertuning for logistic regression
lg=LogisticRegression()
parameters={'penalty':['l1','l2','elasticnet'],'class_weight':['dict','balanced']}

clf=GridSearchCV(lg,parameters)
clf.fit(x_train[0:2001],y_train[0:2001])  # subset of train data
print(clf.best_params_)
```

```
{'class_weight': 'dict', 'penalty': 'l2'}
```

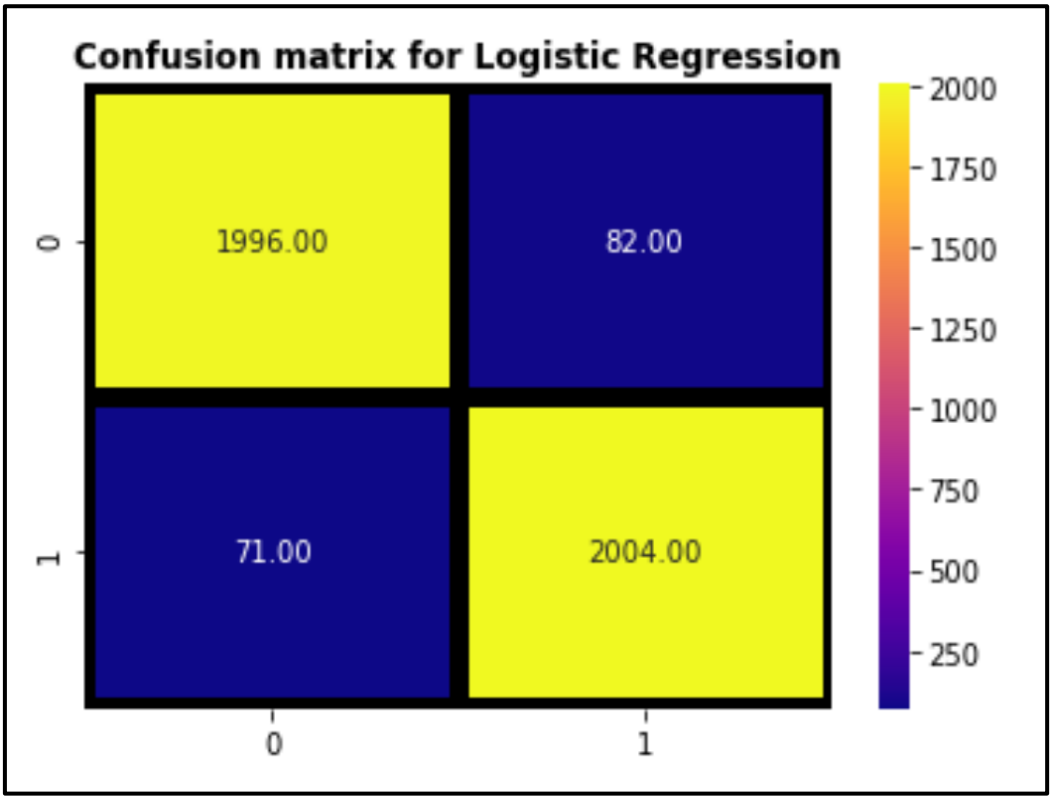
- **Implementing logistic regression with above hyper parameters**

```
lg=LogisticRegression(penalty='l2',class_weight='dict')
lg.fit(x_train,y_train)
lg_pred = lg.predict(x_test) #  predictions

#accuracy and cross validation scores
lg_accuracy= accuracy_score(y_test,lg_pred)*100
print(" Accuracy of logistic regression is  = ",lg_accuracy)
lg_cv=cross_val_score(lg,x,y,cv=3).mean()*100
print("cross validation score=",lg_cv)
print("\n")

#auc
lg_auc = np.round(roc_auc_score(y_test,lg_pred), 3)
print("Auc for logistic regression is {}". format(lg_auc))
print("\n")

# classification report and confusion matrix
print("Confusion matrix \n ",confusion_matrix(y_test,lg_pred))
print("\n")
print("classification report for logistic\n\n",classification_report(y_test,lg_pred))
```

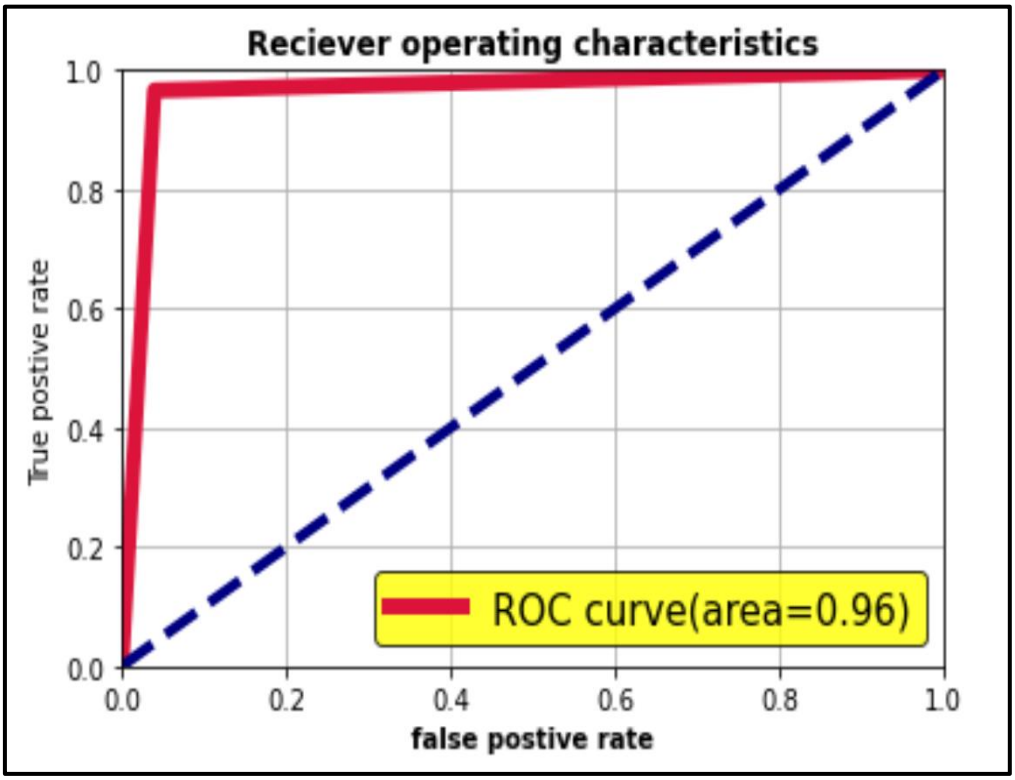
- **Evaluation metrics of logistic Regression**

```
 Accuracy of logistic regression is  =  96.3159162051529
cross validation score= 95.72757536959226


Auc for logistic regression is 0.963


Confusion matrix
  [[1996   82]
 [  71 2004]]


classification report for logistic

              precision    recall  f1-score   support

           0       0.97      0.96      0.96      2078
           1       0.96      0.97      0.96      2075

    accuracy                           0.96      4153
   macro avg       0.96      0.96      0.96      4153
weighted avg       0.96      0.96      0.96      4153
```

- **Confusion matrix for logistic regression**



- **Auc Roc curve**



2) Passive Aggressive classifier

- **Hyper parameter tuning**

```
#hypertuning for passive aggressive classifier
pac=PassiveAggressiveClassifier()
parameters={'fit_intercept':[True,False],'max_iter':[1000,2000,3000],
            'loss':['hinge','squared_hinge'],'warm_start':[False,True],
            'class_weight':[None,'balanced']}
clf=GridSearchCV(pac,parameters)
clf.fit(x_train[0:2001],y_train[0:2001])
print(clf.best_params_)
```

```
{'class_weight': None, 'fit_intercept': True, 'loss': 'hinge', 'max_iter': 1000, 'warm_start': True}
```

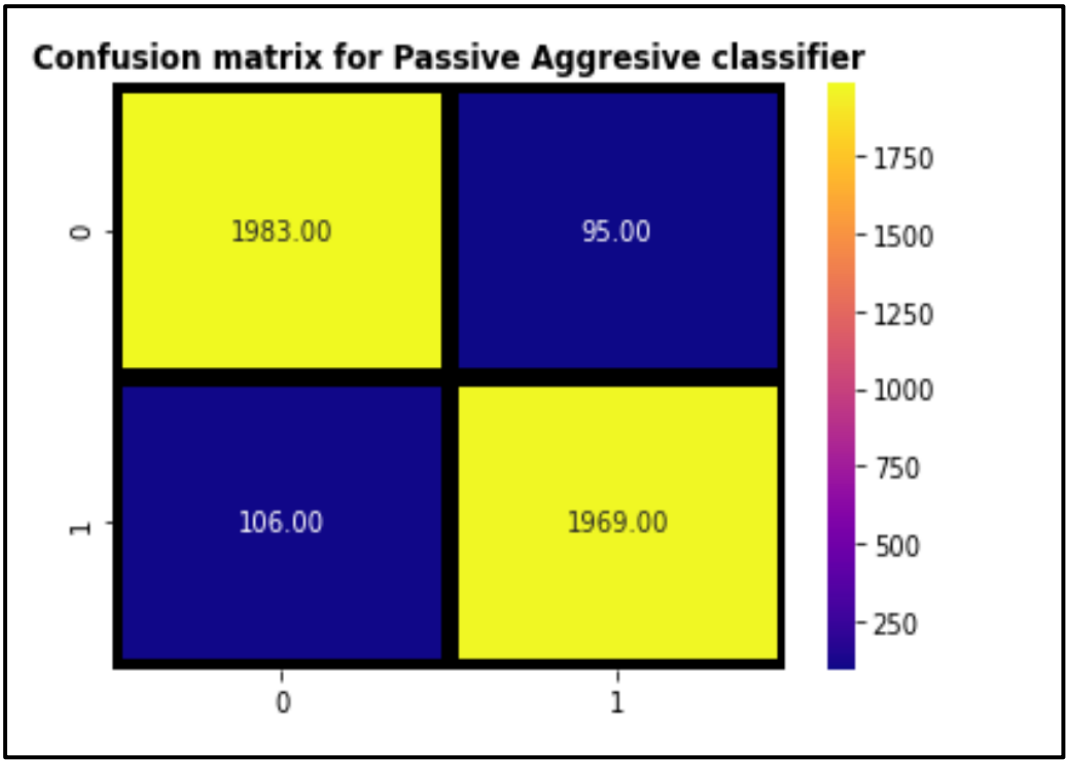
- **Implementing passive aggressive classifier with above hyper parameters**

```
pac=PassiveAggressiveClassifier(loss='hinge',fit_intercept=True,max_iter=3000,warm_start=True,class_weight=None ,shuffle=True,
pac.fit(x_train,y_train)
pac_pred = pac.predict(x_test)

#accuracy and cross validation score
pac_accuracy=accuracy_score(y_test,pac_pred)*100
print(" Accuracy of passive aggressive classifier is = ", pac_accuracy)
pac_cv=cross_val_score(pac,x,y,cv=3).mean()*100
print("cross validation score=",pac_cv)
print("\n")

#auc
pac_auc = np.round(roc_auc_score(y_test,pac_pred), 3)
print("Auc for passive aggresive classifier is {}". format(pac_auc))
print("\n")

#classification report and confusion matrix
print("Confusion matrix \n\n ",confusion_matrix(y_test,pac_pred))
print("\n")
print("classification report for passive aggresive classifier \n\n",classification_report(y_test,pac_pred))
```

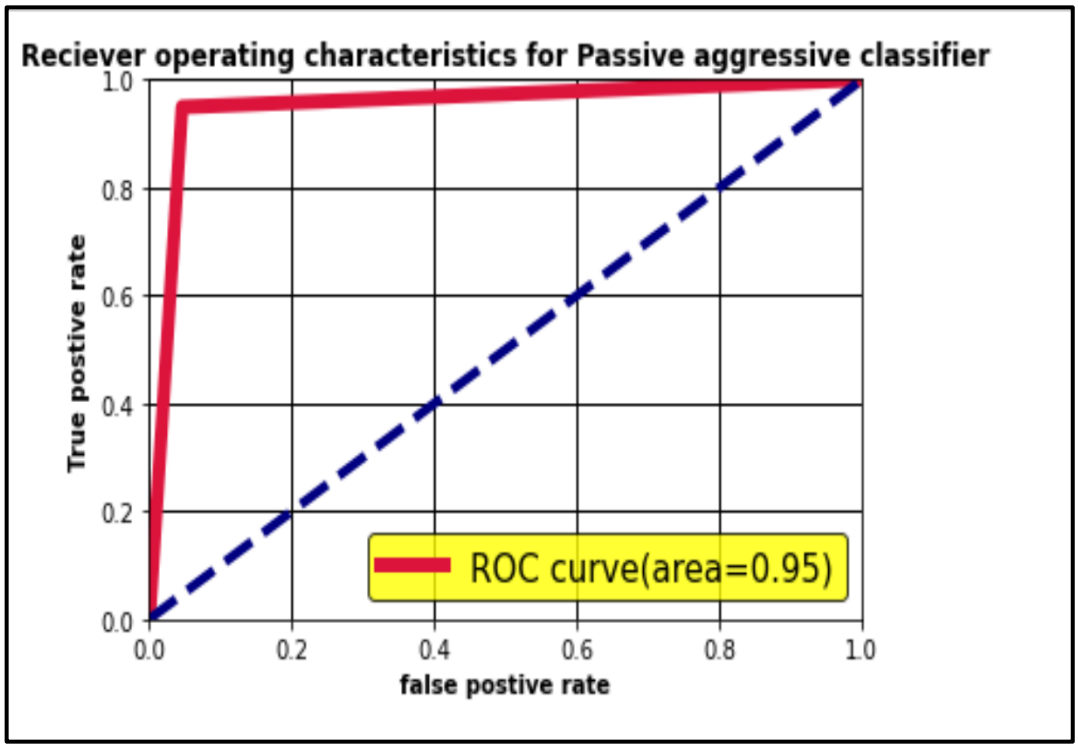
- **Evaluation metrics of passive aggressive classifier**

```
 Accuracy of passive aggressive classifier is =  95.16012521069108
cross validation score= 94.89909239952462


Auc for passive aggresive classifier is 0.952


Confusion matrix

  [[1983   95]
 [ 106 1969]]


classification report for passive aggresive classifier

              precision    recall  f1-score   support

           0       0.95      0.95      0.95      2078
           1       0.95      0.95      0.95      2075

    accuracy                           0.95      4153
   macro avg       0.95      0.95      0.95      4153
weighted avg       0.95      0.95      0.95      4153
```

- **Confusion matrix for passive aggressive classifier**



- **Auc roc curve for passive aggressive classifier**



3) KNeighbors classifier

- Hyper parameter tuning

```
#hypertunning KNN Classifier
knn=KNeighborsClassifier()
parameters={"n_neighbors":[4,6,8],"weights":['uniform','distance'],"algorithm":["auto","ball_tree","brute"]}
abc=GridSearchCV(knn,parameters)
abc.fit(x_train[0:2001],y_train[0:2001])
print(abc.best_params_)
```

```
{'algorithm': 'auto', 'n_neighbors': 8, 'weights': 'distance'}
```

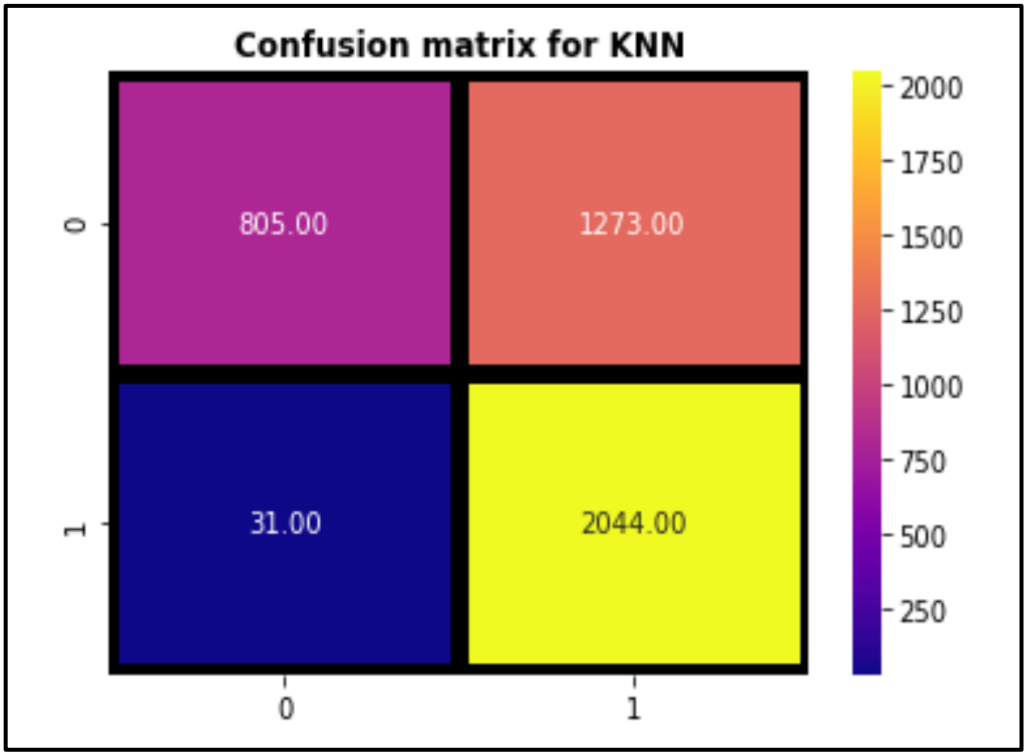
- **Implementing KNeighbors classifier with above hyper parameters**

```
knn=KNeighborsClassifier(algorithm='auto',n_neighbors=8,weights='distance')
knn.fit(x_train,y_train)
knn_pred=knn.predict(x_test)

#accuracy and cross validation score
knn_accuracy=accuracy_score(y_test,knn_pred)*100
print(" Accuracy of KNN classifier is = ",knn_accuracy)
knn_cv=cross_val_score(knn,x,y,cv=3).mean()*100
print("cross_validation score=",knn_cv)
print("\n")

# AUC
knn_auc = np.round(roc_auc_score(y_test,knn_pred), 3)
print("Auc for KNN classifier is {}". format(knn_auc))
print("\n")


#classification report and confusion matrix
print("confusion matrix \n\n",confusion_matrix(y_test,knn_pred))
print("\n")
print("classification report of KNN classifier\n\n",classification_report(y_test,knn_pred))
```

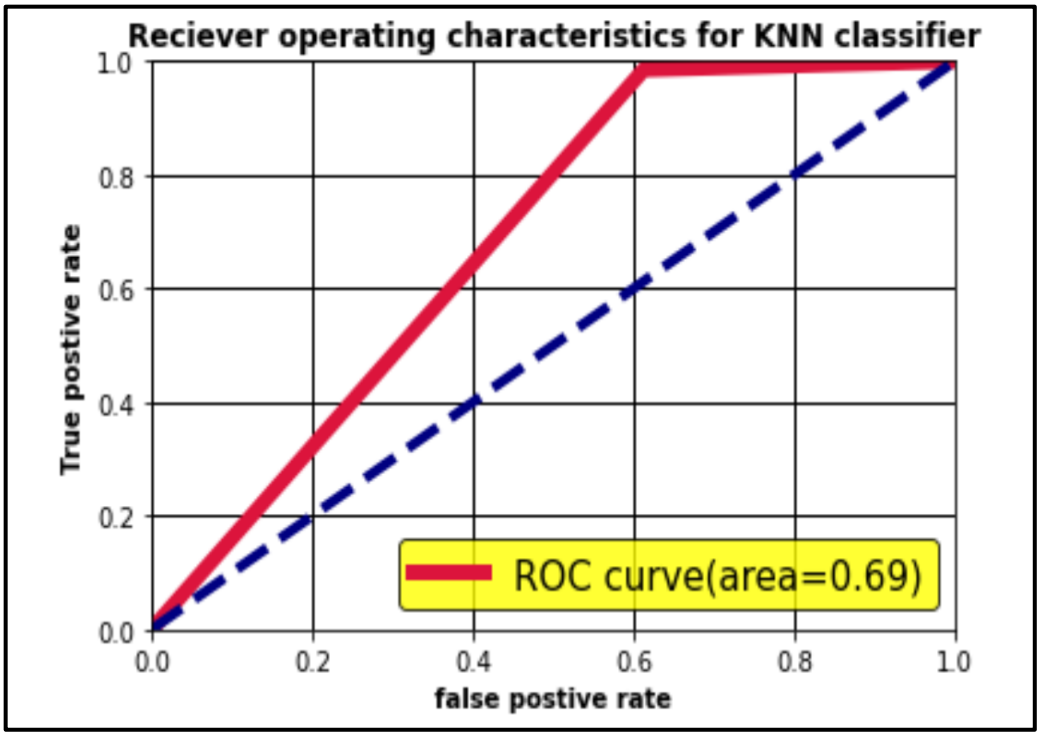
- **Evaluation metrics of KNeighbors classifier**

```
 Accuracy of KNN classifier is =  68.60101131712015
cross_validation score= 79.04298483936104


Auc for KNN classifier is 0.686


confusion matrix

 [[ 805 1273]
 [  31 2044]]


classification report of KNN classifier

              precision    recall  f1-score   support

           0       0.96      0.39      0.55      2078
           1       0.62      0.99      0.76      2075

    accuracy                           0.69      4153
   macro avg       0.79      0.69      0.66      4153
weighted avg       0.79      0.69      0.66      4153
```

- **Confusion matrix for KNeighbors classifier**



- **Auc roc curve for KNeighbors classifier**



4) Multinomial NB classifier

- **Hyper parameter tuning**

```
#hypertuning for multinomial
MNB=MultinomialNB()
parameters={"alpha":[0.0,1.0],"fit_prior":[True,False]}

clf=GridSearchCV(MNB,parameters)
clf.fit(x_train[0:2001],y_train[0:2001])
print(clf.best_params_)
```

```
{'alpha': 1.0, 'fit_prior': False}
```

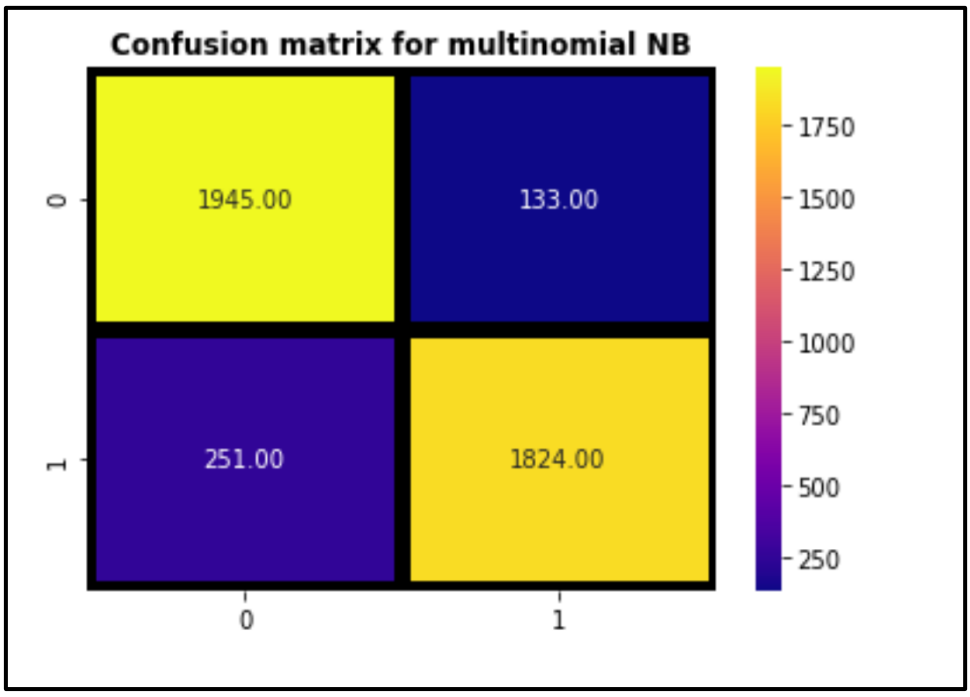
- **Implementing Multinomial NB classifier with above hyper parameters**

```
MNB=MultinomialNB(alpha=1.0,fit_prior=False)
MNB.fit(x_train,y_train)
MNB_pred = MNB.predict(x_test) #  predictions

#accuracy and cross validation score
MNB_accuracy=accuracy_score(y_test,MNB_pred)*100
print(" Accuracy of Multinomial nb classifier is = ", MNB_accuracy)
MNB_cv=cross_val_score(MNB,x,y,cv=3).mean()*100
print("cross validation score=",MNB_cv)
print("\n")

#auc
MNB_auc = np.round(roc_auc_score(y_test,MNB_pred), 3)
print("Auc for Multinomial NB is {}". format(MNB_auc))
print("\n")

#classification report and confusion matrix
print("Confusion matrix \n\n ",confusion_matrix(y_test,MNB_pred))
print("\n")
print("classification report for Multinomial classifier \n\n",classification_report(y_test,MNB_pred))
```

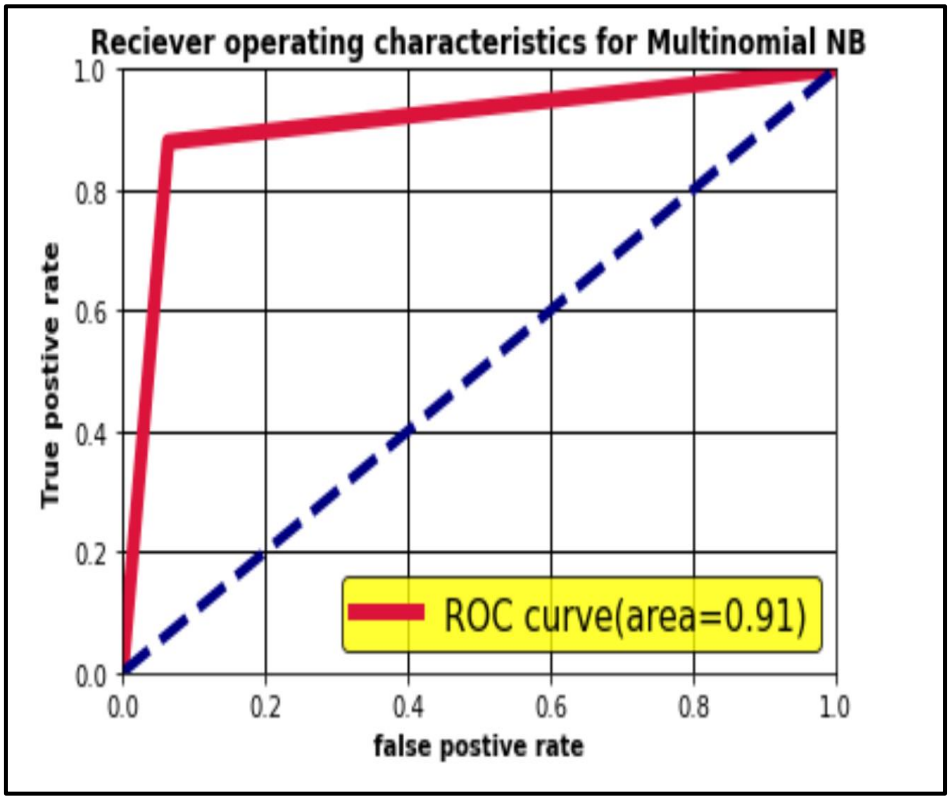
- **Evaluation metrics of Multinomial NB classifier**

```
 Accuracy of Multinomial nb classifier is =  90.75367204430532
cross validation score= 90.1979719370746


Auc for Multinomial NB is 0.908


Confusion matrix

  [[1945  133]
 [ 251 1824]]


classification report for Multinomial classifier

              precision    recall  f1-score   support

           0       0.89      0.94      0.91      2078
           1       0.93      0.88      0.90      2075

    accuracy                           0.91      4153
   macro avg       0.91      0.91      0.91      4153
weighted avg       0.91      0.91      0.91      4153
```

- **Confusion matrix for Multinomial NB classifier**



- **Auc roc curve for Multinomial NB classifier**



5) SVC

- **Hyper parameter tuning**

```
#hypertuning for svc
from sklearn.model_selection import GridSearchCV
svc=SVC()
parameters={"kernel":["linear","poly","rbf"],"gamma":["scale","auto"]}

clf=GridSearchCV(svc,parameters)
clf.fit(x_train[0:2001],y_train[0:2001])
print(clf.best_params_)
```

```
{'gamma': 'scale', 'kernel': 'linear'}
```

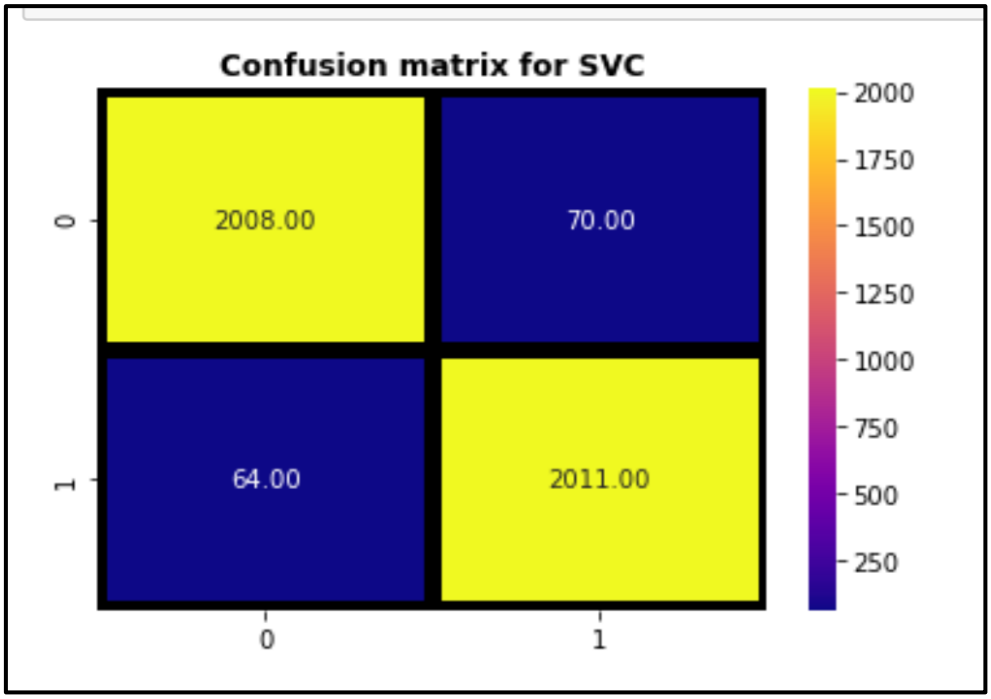
- **Implementing SVC with above hyper parameters**

```
#using the best parameter for svc from the one's obtained above
svc=SVC(kernel='linear',gamma="scale")
svc.fit(x_train,y_train)
svc_pred=svc.predict(x_test)

#accuracy and cross validation score
svc_accuracy=accuracy_score(y_test,svc_pred)*100
print("Accuracy of SVC =",svc_accuracy)
svc_cv=cross_val_score(svc,x,y,cv=3).mean()*100
print("cross_validation score=",svc_cv)
print("\n")

#auc
svc_auc = np.round(roc_auc_score(y_test,svc_pred), 3)
print("Auc for svc is {}". format(svc_auc))
print("\n")

#classification report and confusion matrix
print("confusion matrix \n\n",confusion_matrix(y_test,svc_pred))
print("\n")
print("classification report of SVC \n\n",classification_report(y_test,svc_pred))
```

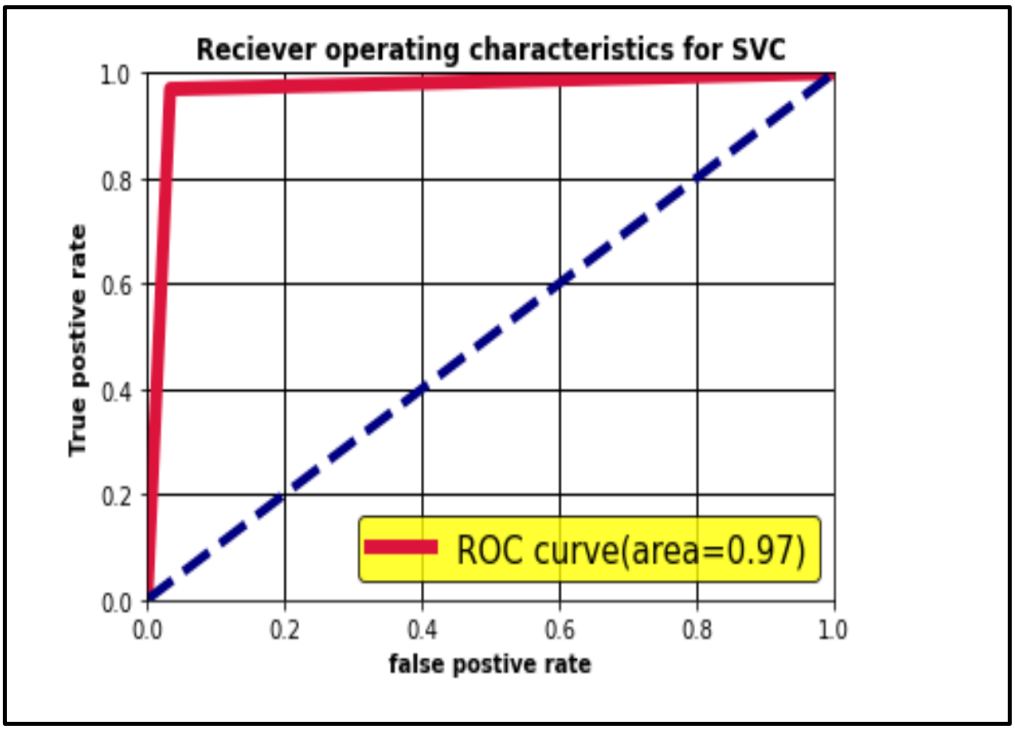
- **Evaluation metrics of SVC**

```
Accuracy of SVC = 96.77341680712738
cross_validation score= 96.267032646724


Auc for svc is 0.968


confusion matrix

 [[2008   70]
 [  64 2011]]


classification report of SVC

                precision    recall  f1-score   support

           0       0.97      0.97      0.97      2078
           1       0.97      0.97      0.97      2075

    accuracy                           0.97      4153
   macro avg       0.97      0.97      0.97      4153
weighted avg       0.97      0.97      0.97      4153
```

- **Confusion matrix for SVC**



- **Auc roc curve for SVC**



6) Gradient Boosting classifier

- **Hyper parameter tuning Gradient Boosting classifier**

```
#hypertuning for gradient boosting classifier
gb=GradientBoostingClassifier()
parameters={"n_estimators":[10,20,30],"criterion":["mae","mse"],'max_features':['auto','sqrt','log2'],'loss':['deviance','exponer

clf=GridSearchCV(gb,parameters)
clf.fit(x_train[0:1001],y_train[0:1001])
print(clf.best_params_)
```

```
{'criterion': 'mse', 'loss': 'deviance', 'max_features': 'auto', 'n_estimators': 30}
```

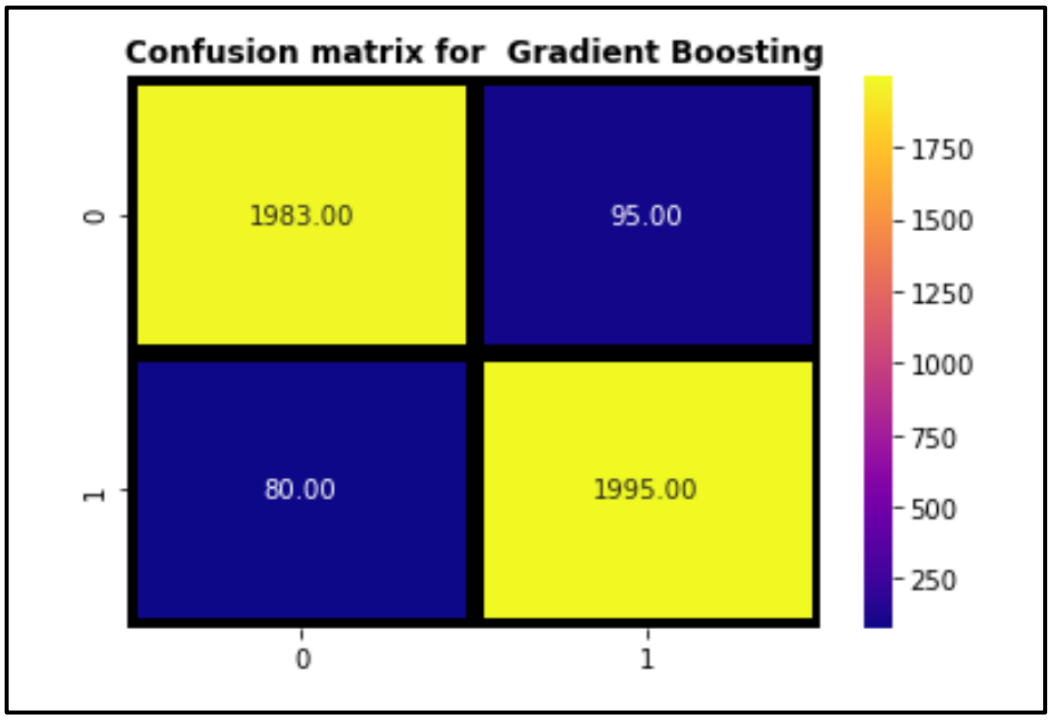
- **Implementing Gradient Boosting classifier with above hyper parameters**

```
#using the best parameter obtained above
gb=GradientBoostingClassifier(criterion="mse",n_estimators=30,max_features="auto",loss="deviance")
gb.fit(x_train,y_train)
gb_pred=gb.predict(x_test)

gb_accuracy=accuracy_score(y_test,gb_pred)*100
print("Accuracy of gradient boosting classifier",gb_accuracy)
gb_cv=cross_val_score(gb,x,y,cv=3).mean()*100
print("cross_validation score=",gb_cv)
print("\n")

#auc
gb_auc = np.round(roc_auc_score(y_test,gb_pred), 3)
print("Auc for gradient boosting is {}". format(gb_auc))
print("\n")

print("confusion matrix \n\n",confusion_matrix(y_test,gb_pred))
print("\n")
print("classification report of Gradient Boosting classifier \n\n",classification_report(y_test,gb_pred))
```

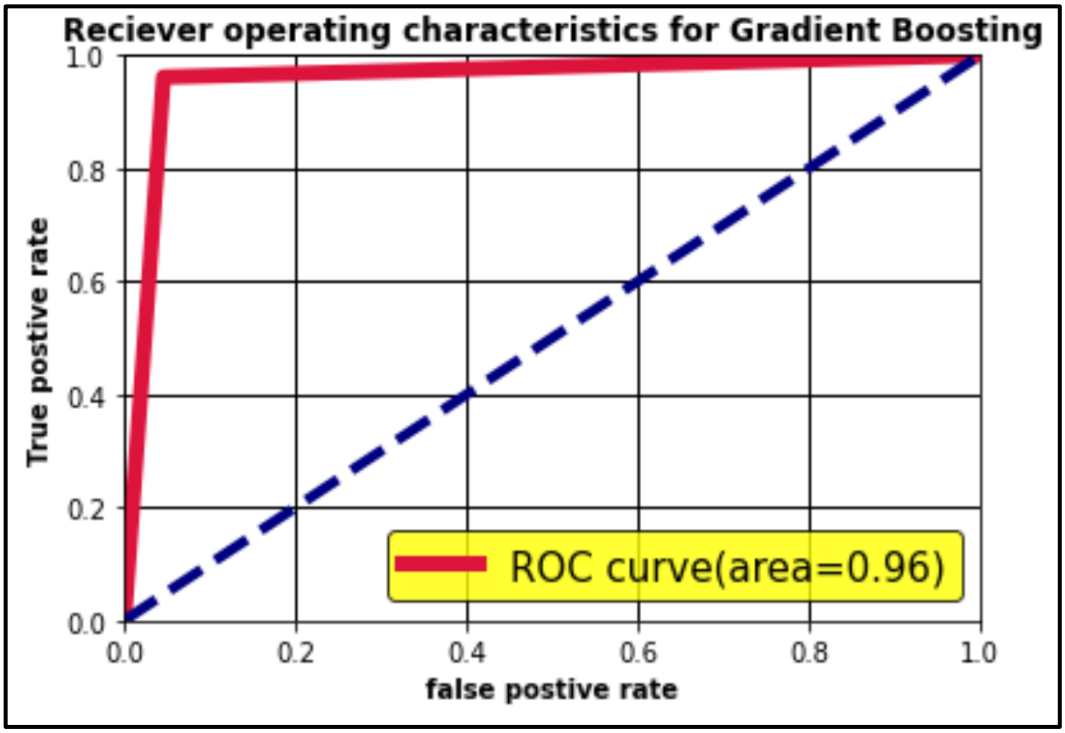
- **Evaluation metrics of Gradient Boosting classifier**

```
Accuracy of gradient boosting classifier 95.78617866602455
cross_validation score= 95.63122651203409


Auc for gradient boosting is 0.958


confusion matrix

 [[1983   95]
 [  80 1995]]


classification report of Gradient Boosting classifier

              precision    recall  f1-score   support

           0       0.96      0.95      0.96      2078
           1       0.95      0.96      0.96      2075

    accuracy                           0.96      4153
   macro avg       0.96      0.96      0.96      4153
weighted avg       0.96      0.96      0.96      4153
```

- **Confusion matrix for Gradient Boosting classifier**



- **Auc roc curve for Gradient Boosting classifier**



- ## Key Metrics for success in solving problem under consideration

Choosing the right metric is crucial while evaluating machine learning (ML) models. Various metrics are proposed to evaluate ML models in different applications .In some applications looking at a single metric may not give you the whole picture of the problem you are solving, and you may want to use a subset of the metrics.

There are various ways to evaluate a classification model, and we will be covering some of the most popular ones below.

1- Confusion Matrix

One of the key concept in classification performance is confusion matrix (AKA error matrix), which is a tabular visualization of the model predictions versus the ground-truth labels. Each row of confusion matrix represents the instances in a predicted class and each column represents the instances in an actual class. Confusion Matrix is a 2X2 Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Understanding True Positive, True Negative, False Positive and False Negative in a Confusion Matrix

1) **True Positive (TP)**
- The predicted value matches the actual value
- The actual value was positive and the model predicted a positive value

2) **True Negative (TN)**
- The predicted value matches the actual value.
- The actual value was negative and the model predicted a negative value

3) **False Positive (FP) – Type 1 error**
- The predicted value was falsely predicted

- The actual value was negative but the model predicted a positive value
- Also known as the Type 1 error

4) False Negative (FN) – Type 2 error

- The predicted value was falsely predicted
- The actual value was positive but the model predicted a negative value
- Also known as the Type 2 error

Let’s go through this with an example. Let’s assume we are building a binary classification . There are two possible predicted classes: "yes" and "no". If we were predicting the presence of a disease, for example, "yes" would mean they have the disease, and "no" would mean they don't have the disease.

The classifier made a total of 165 predictions (e.g., 165 patients were being tested for the presence of that disease).

Out of those 165 cases, the classifier predicted "yes" 110 times, and "no" 55 times.

In reality, 105 patients in the sample have the disease, and 60 patients do not.

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

Let's now define the most basic terms, which are whole numbers (not rates):

- true positives (TP): These are cases in which we predicted yes (they have the disease), and they do have the disease.
- True negatives (TN): We predicted no, and they don't have the disease.
- False positives (FP): We predicted yes, but they don't actually have the disease. (Also known as a "Type I error.")
- False negatives (FN): We predicted no, but they actually do have the disease. (Also known as a "Type II error.")

2- Classification Accuracy

Classification accuracy is perhaps the simplest metrics one can imagine, and is defined as the number of correct predictions divided by the total number of predictions, multiplied by 100. For an example if out of 1100 samples 1030 are predicted correctly, resulting in a classification accuracy of:

Classification accuracy=1030/1100= 93.6%

3- Precision

There are many cases in which classification accuracy is not a good indicator of your model performance. One of these scenarios is when your class distribution is imbalanced (one class is more frequent than others). In this case, even if you predict all samples as the most frequent class you would get a high accuracy rate, which does not make sense at all (because your model is not learning anything, and is just predicting everything as the top class.

Therefore we need to look at class specific performance metrics too. Precision is one of such metrics, which is defined as:

Precision= True Positive/ (True Positive+ False Positive)

$$Precision = \frac{TP}{TP + FP}$$

Precision tells us how many of the correctly predicted cases actually turned out to be positive.

4- Recall

Recall is another important metric, which is defined as the fraction of samples from a class which are correctly predicted by the model. More formally:

Recall= True Positive/ (True_Positive+ False_Negative)

$$Recall = \frac{TP}{TP + FN}$$

Recall tells us how many of the actual positive cases we were able to predict correctly with our model.

Note:

- Precision is a useful metric in cases where False Positive is a higher concern than False Negatives.
- Recall is a useful metric in cases where False Negative trumps False Positive.

5- F1 Score

Depending on application, you may want to give higher priority to recall or precision. But there are many applications in which both recall and precision are important. Therefore, it is natural to think of a way to combine these two into a single metric. One popular metric which combines precision and recall is called F1-score, which is the harmonic mean of precision and recall defined as:

F1-score= 2*Precision*Recall/(Precision + Recall)

$$F_\beta = (1+\beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}}$$

It is good to mention that there is always a trade-off between precision and recall of a model, if you want to make the precision too high, you would end up seeing a drop in the recall rate, and vice versa.

F1-score is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. It is maximum when Precision is equal to Recall.

6- Sensitivity and Specificity

Sensitivity and specificity are two other popular metrics mostly used in medical and biology related fields, and are defined as:

Sensitivity= Recall= TP/(TP+FN)

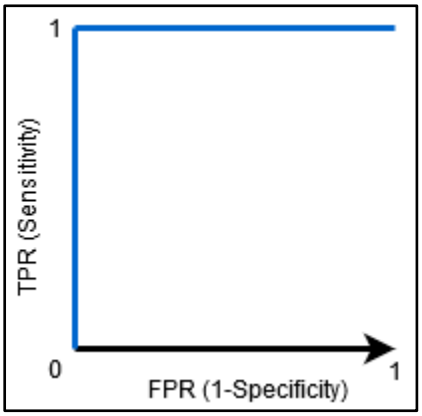
Specificity= True Negative Rate= TN/(TN+FP)

7-AUC-ROC curve

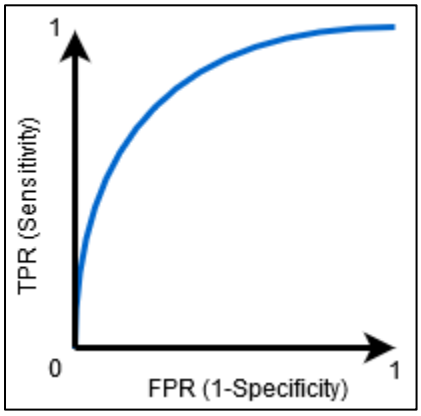
The Receiver Operator Characteristic (ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise'.

The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve.

The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

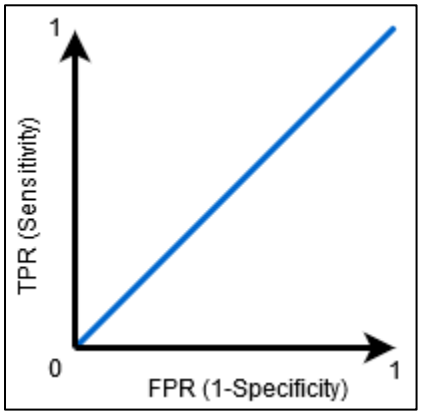


When AUC = 1, then the classifier is able to perfectly distinguish between all the Positive and the Negative class points correctly. If, however, the AUC had been 0, then the classifier would be predicting all Negatives as Positives, and all Positives as Negatives.



When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative class values. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives.

When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative class values. This is so because the classifier is able to detect more numbers of True positives and True negatives than False negatives and False positives.



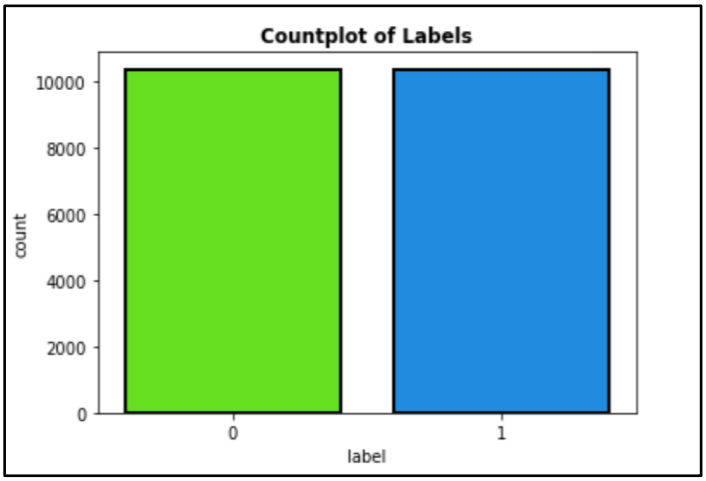
When AUC=0.5, then the classifier is not able to distinguish between Positive and Negative class points. Meaning either the classifier is predicting random class or constant class for all the data points.

So, the higher the AUC value for a classifier, the better its ability to distinguish between positive and negative classes.

- # Visualizations

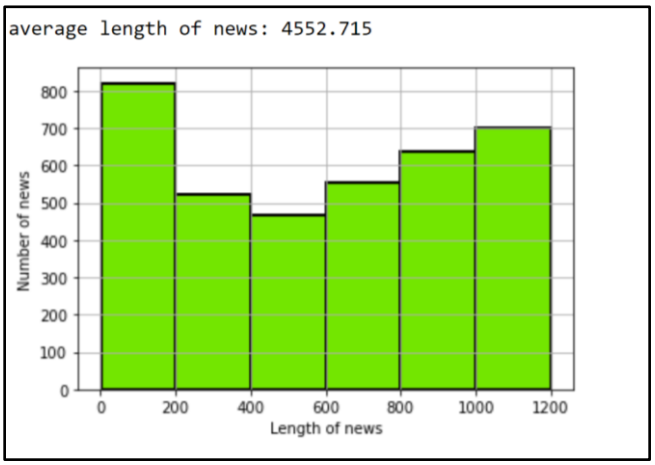
  - Data analysis and data visualization are not entirely exclusive work sets – they co-exist and feed off of each other.
  - Thus data visualization aids in gaining useful insights from large data and makes it easy to understand.
  - In the journey from analysis to data-driven outcomes, data visualization plays a very important role of presenting data in a powerful and credible way.
  - There are many different scenarios where large amounts of data must be displayed to an audience – a business may need to present sales figures to their directors.
  - A research team may need to display their findings to investors, or a teacher may need to display statistics to their students for example.
  - Data visualization allows this information to be displayed in an easy to read format that is both attractive and functional.
  - Let us see some of the visualizations carried out in our project

    1) Count plot of the Label:



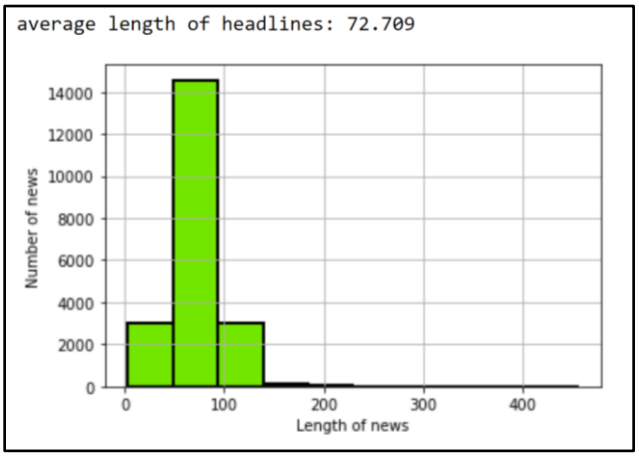
- There are 10374 records for fake news(1)
- And 10387 real news(0)
- Thus we can say the classes were balanced.

2) Average length of news



- The average numbers of words in the news column were concentrated at 4552.
- Majority of the news had words from 0 to 200

3) Average length headlines



- The average numbers of words in the headline column were 72
- Majority of the headlines had words from 100 to 200 only

Word Clouds

- Many times you might have seen a cloud filled with lots of words in different sizes, which represent the frequency or the importance of each word. This is called Tag Cloud or WordCloud.

- WordCloud is a technique to show which words are the most frequent among the given text. The size of the words represents their frequency. Wordcloud tool is quite handy for exploring text data and making your report livelier.

- Each word in this cloud has a variable font size and colour tone. Thus, this representation helps to determine words of prominence. A bigger font size of a word portrays its prominence more relative to other words in the cluster.

- Word Cloud can be built in varying shapes and sizes based on the creators' vision. The number of words plays an important role while creating a Word Cloud. More number of words does not always mean a better Word Cloud as it becomes clustery and difficult to read.

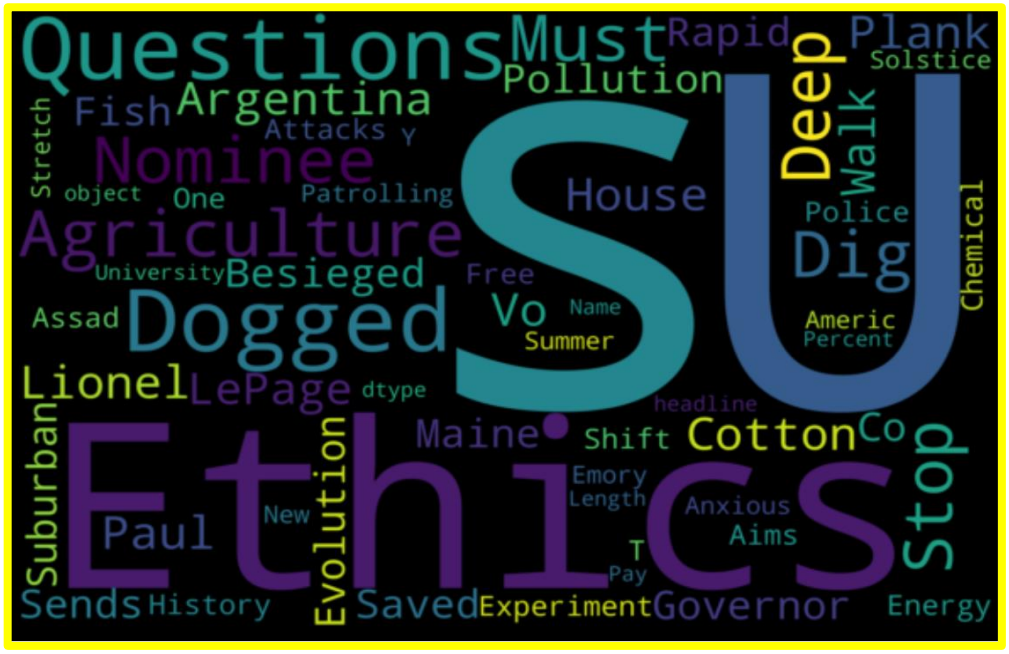
- A Word Cloud must always be semantically meaningful and must represent what it is meant for.

1) Word cloud for all Headlines

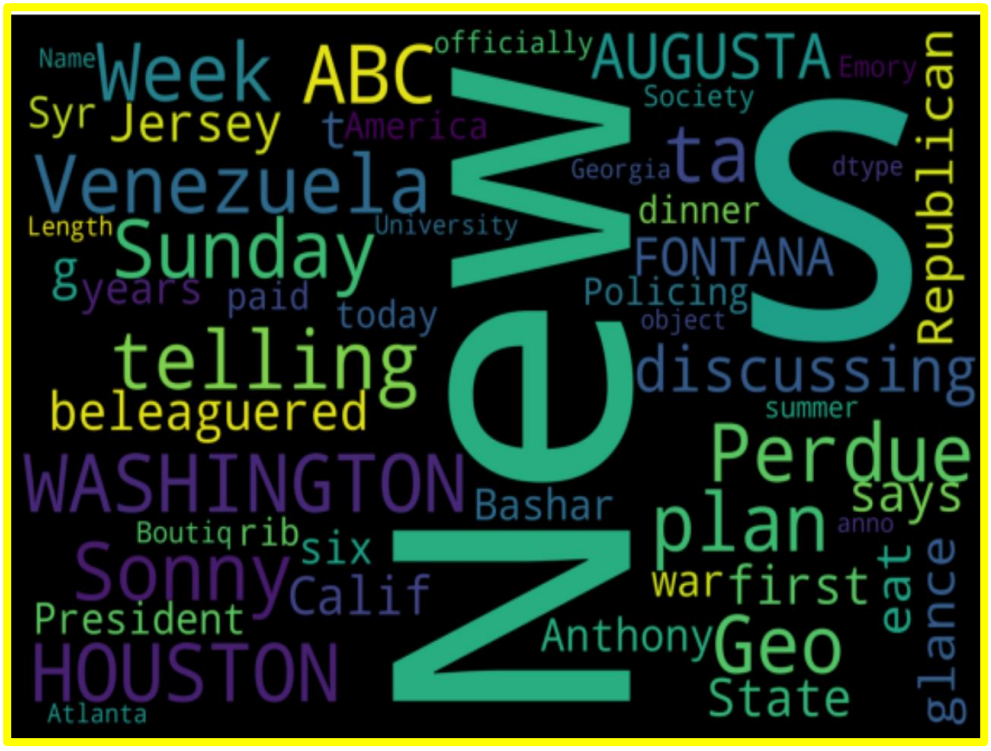
The loud words in headlines were US, ethics, Agriculture, questions Argentina Etc.

This shows most of the news included US as its topic and were mostly concerned with politics as we can see words like Trump, Lionel, House, Governor etc.

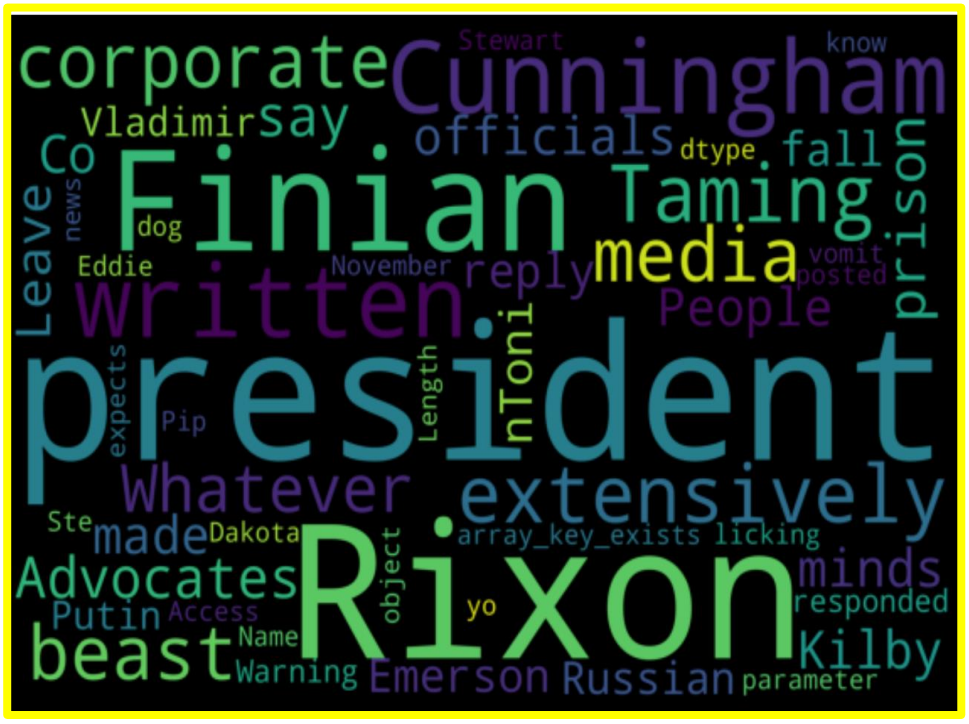
2) Word cloud for headlines labelled as true



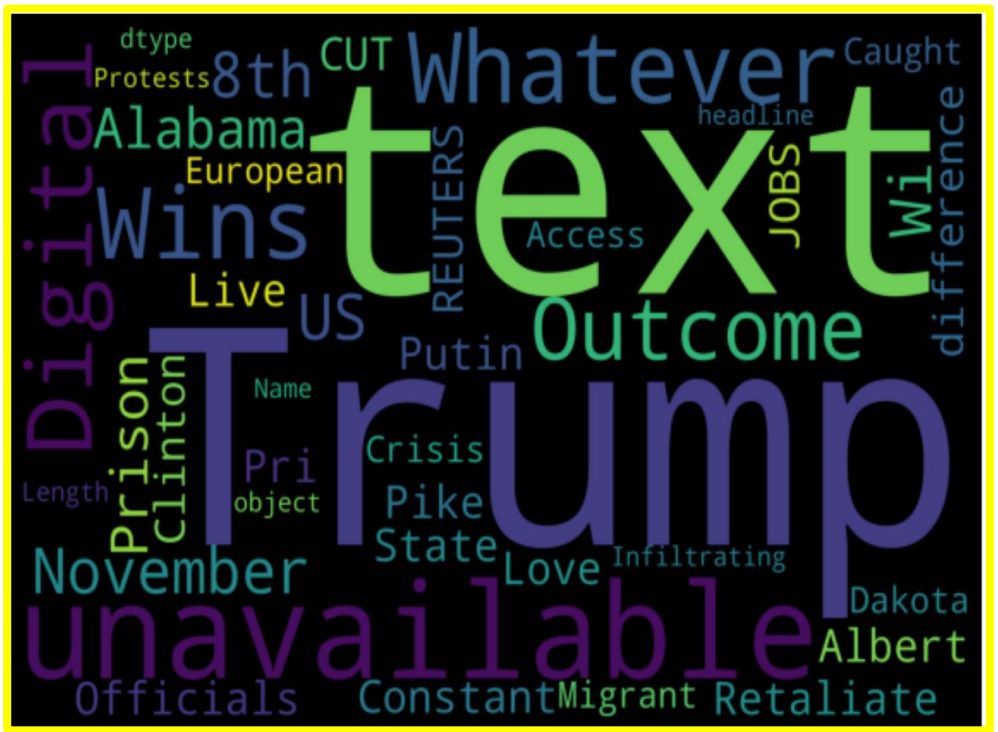
3) Word cloud for news labelled as true



4) **Word cloud for fake news**



5) **Word cloud for fake headlines**



- ## Interpretation of the Results

We have implemented the following algorithms:

1) Logistic Regression
2) Passive aggressive classifier
3) KNN
4) Multinomial
5) SVC
6) Gradient Boosting classifier

The evaluation metrics used were accuracy score, cross validation, precision, recall, auc roc scores and the confusion matrix.

The learning algorithms are trained with different hyper parameters to achieve maximum accuracy for a given dataset, with an optimal balance between variance and bias. Each model is trained multiple times with a set of different parameters using a grid search to optimize the model for the best outcome. Using a grid search to find the best parameters is computationally expensive however, the measure is taken to ensure the models do not over fit or under fit the data.

As the data size was big and my system took forever to hyper parameter tune I have used only a subset of train data to get the hyper parameters and then the final model was implemented.

After all the exploratory data analysis and text pre-processing with 20 % test size the model was put to train and predictions were obtained.

Here is the brief overview of how well the various algorithms are working:

	Model	Accuracy Score	Cross Validation score	AUC-ROC
0	SVC	96.773417	96.267033	0.968
1	Logistic Regression	96.315916	95.727575	0.963
2	Gradient Boosting	95.786179	95.631227	0.958
3	Passive Aggressive classifier	95.160125	94.899092	0.952
4	Multinomial	90.753672	90.197972	0.908
5	KNN	68.601011	79.042985	0.686

When it comes to classification problem it becomes necessary to study the precision and recall values as it will indicate how well the model will classify all the classes

present in the dataset. Hence only evaluating the accuracy will not help us .We shall consider various evaluation metrics before coming to a conclusion of perfect model

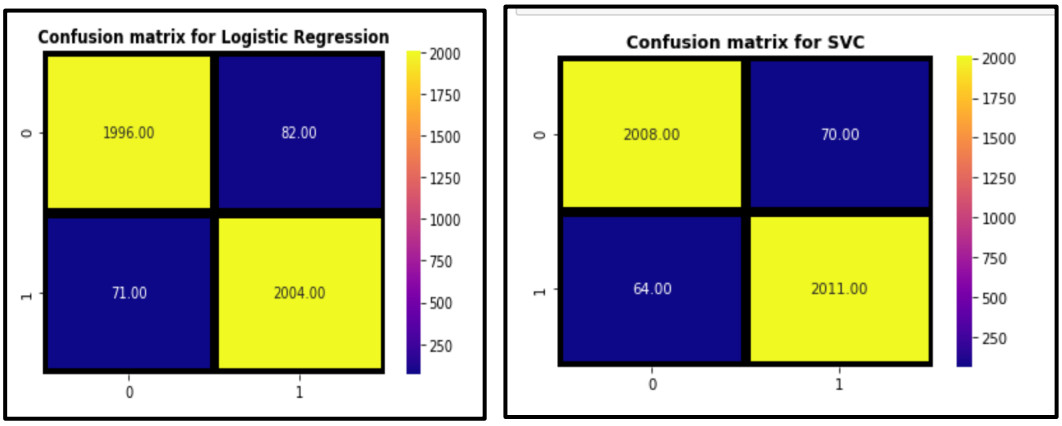
Looking at the highest accuracy, SVC is giving us the highest accuracy of 96.77 which is validated with a cross validation of 96.26 which says it does over fit the model.

Logistic Regression could be seen giving similar accuracy to that of SVC but logistic regression shows very little over fit as the accuracy score is slightly greater than cross validation.

But there was a difference in precision and recall values of logistic and Svc.

Both precision and recall for SVC is 0.97 while for logistic regression it is 0.96 for one class and 0.97 for the other, So we choose SVC over logistic here.

Taking confusion matrix into consideration. Let us have a quick glance at both the confusion matrix



The true positives for logistic are 1996 while for Svc are 2008

The true negatives for logistic regression are 2004 while for SVC there are 2011 true negatives. The total numbers of errors for Logistic are 153 while for SVC there are 124 errors.

Hence we used confusion matrix as a deciding factor between Logistic regression and SVC and finalized SVC as our best working model.

Among all the six algorithms KNN had very poor evaluation metrics with an accuracy of only 68.60 % and cross validation of 79% which states the model was over fitting also the precision and recall values were unsatisfactory.

The Auc Roc score was KNeighbors classifier was as low as 0.69.

Rest of the algorithms less or more worked similarly showing no over-fitting or under fitting and auc roc scores above 0.90 which was again satisfactory.

CONCLUSION

• Key Findings and Conclusions of the Study

- Advances in technology and the spread of news through different types of media have increased the spread of fake news today. One of the principal problems is that receivers believe anything sent to them over social media due to lack of awareness.

- The task of classifying news manually requires in-depth knowledge of the domain and expertise to identify anomalies in the text.

- In the discourse of not being able to detect fake news, the world would no longer hold value in truth.

- These fake news can be propaganda against an individual, society, organization or political party. Information veracity is a long-term issue affecting society both for printed and digital media.

- A human being is unable to detect all these fake news. So there is a need for machine learning classifiers that can detect this fake news automatically.

- Hence the problem has been taken over and resolved with the help of Natural Language Processing tools which help us identify fake or true news based on historical data. The news is now in safe hands!

- The main contribution of this project is support for the idea that machine learning could be useful in a novel way for the task of classifying fake news.

- This project work makes an analysis of the research related to fake news detection and explores the traditional machine learning models to choose the best, in order to create a model of a product with supervised machine learning algorithm, that can classify fake news as true or false, by using tools like python scikit-learn, NLP for textual analysis.

- In this report, we discussed the problem of classifying fake news articles using machine learning models and few ensemble techniques.

- The primary aim was to identify patterns in text that differentiate fake articles from true news. We extracted different textual features from the articles using a Tfidf and used the feature set as an input to the models.

- The learning models were trained and parameter-tuned to obtain optimal accuracy and achieve an optimal balance between variance and bias. Some models have achieved comparatively higher accuracy than others.

- Each model is trained multiple times with a set of different parameters using a grid search to optimize the model for the best outcome. Using a grid search to find the best parameters is computationally expensive however, the measure is taken to ensure the models do not over fit or under fit the data.

- We used multiple performance metrics to compare the results for each algorithm. The SVC has shown an overall better score on all performance metrics as compared to others.

- Based on the recall and precision performance metric, SVC classifier stands best by achieving a recall score of 0.97. This is closely followed by Gradient boosting classifier) and logistic regression which achieved a recall of 0.96.

- SVC is giving us the highest accuracy of 96.77 which is validated with a cross validation of 96.26 which says it does over fit the model.

- Logistic Regression could be seen giving similar accuracy to that of SVC but logistic regression shows very little over fit as the accuracy score is slightly greater than cross validation.

- The true positives for logistic are 1996 while for Svc are 2008.The true negatives for logistic regression are 2004 while for SVC there are 2011 true negatives.

- The total numbers of errors for Logistic are 153 while for SVC there are 124 errors. Hence we used confusion matrix as a deciding factor between Logistic regression and SVC and finalized SVC as our best working model.

- Among all the six algorithms KNN had very poor evaluation metrics with an accuracy of only 68.60 % and cross validation of 79% which states the model was over fitting also the precision and recall values were unsatisfactory.

- Therefore SVC was finalized and saved for future predictions

- By classifying fake news, we hope to get one step closer towards building an automated fake news detection platform. This study provides a baseline for the future tests and broadens scope of the solutions dealing with fake news detection

- ## Learning Outcomes of the Study in respect of Data Science

Internet is one of the important inventions and a large number of persons are its users. These persons use this for different purposes. There are different social media platforms that are accessible to these users. Any user can make a post or spread the news through these online platforms. These platforms do not verify the users or their posts. So some of the users try to spread fake news through these platforms. These fake news can be a propaganda against an individual, society, organization or political party. A human being is unable to detect all these fake news. So there is a need for machine learning classifiers that can detect these fake news automatically. Use of machine learning classifiers for detecting the fake news.

Increasing use of internet has made it easy to spread the false news. Different social media platforms can be used to spread fake news to a number of persons. With the share option of these platforms, the news spread in a fast way. Fake news just not only affects an individual but it can also affect an organization or business. So controlling the fake news is mandatory. A person can know the news is fake only when he knows the complete story of that topic. It is a difficult task because most of the people do not know about the complete story and they just start believing in the fake news without any verification. The question arises here how to control fake news because a person cannot control the fake news. The answer is machine learning. Machine learning can help in detecting the fake news. Through the use of machine learning these fake news can be detected easily and automatically .Once someone will post the fake news, machine learning algorithms will check the contents of the post and will detect it as fake news. Different researchers are trying to find the best machine learning classifier to detect the fake news. Accuracy of the classifier must be considered because if it failed in detecting the fake news then it can be harmful to different persons. The accuracy of the classifier depends on the training of this classifier. A model that is trained in a good way can give more accuracy.

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 features of the language and content only within the source in question, without utilizing any fact checker or knowledge base.

The core task of detecting fake news involves identifying the language (set of words or sentences) which is used to deceive the readers.

The idea of classifying fake news by learning word-level meaning is a very challenging task under the skin.

For an example:

"Have a Beer, It's Good for Your Brain," reported Inc. But you should wait a minute before you grab a pint (or two). The study was done on mice — not people. And the amount of beer was the equivalent of 28 kegs in humans.

The above examples capture the complex nature of detection and classification of fake news. To rightly classify the above types of fake news, our language model needs to understand the subtleties involved in conveying messages through text.

Other challenge was to extract relevant information from the text data so proper data cleaning was required. For the word embedding the chosen technique that is Tfidf vectorizer was quite a challenge to handle as we had 3 textual columns and tfidf could only be passed with a single column. The easiest solution to this was merging the columns and then transforming the entire corpus to useful vectors which gives us new features.

- ## Limitations of this work and Scope for Future Work

Through the work done in this project, we have shown that machine learning certainly does have the capacity to pick up on sometimes subtle language patterns that may be difficult for humans to pick up on.

The possible future work in this project is augmenting and increasing the size of the dataset. I feel that more data would be beneficial in ridding the model of any bias based on specific patterns in the source. There is also question as to whether or not the size of our dataset is sufficient.

Fake news is not something that is new however. As technology evolves and advances over time, the detection of fake news also becomes more challenging as social media continues to dominate our everyday lives and hence accelerating the speed of which fake news travels

Although there is evident success in detection of fake news and posts using various Machine learning approaches. However ever-changing characteristics and features of fake news in social media networks is posing a challenge in categorization of fake news.

As automated detection of fake news is a hard task to accomplish as it requires the model to understand nuances in natural language. Moreover, majority of the existing fake news detection models treat the problem at hand as a binary classification task, which limits model's ability to understand how related or unrelated the reported news is when compared to the real news. To address these gaps, we can use neural network

architecture to accurately predict the stance between a given pair of headline and article body.

Fake news detection is still a challenge even to deep learning methods such as Convolutional Neural Network (CNN), Recurrent neural network (RNN), etc., because the content of fake news is planned in a way it resembles the truth so as to deceive readers; and without cross referencing and fact checking.

We could further dig deep and evaluate the effects of such fake news propagation on the readers and come up with simple techniques for faster prediction.

Thank-you.