FAKE NEWS DETECTION

DATE:	*1/11/2023*
TEAM ID:	*5378*
PROJECT NAME:	*FAKE NEWS DETECTION USING NLP*

INTRODUCTION:

- Fake news can function as propaganda or misinformation, but it always appeals to the emotions of the public and the intent to cover rational responses, analysis, and comparison of information from several sources, encouraging inflammation and outrage and can easily lead to conspiracy theories and partisan biased content that negatively affects.

- The major source of news and data for the public is served by social media and online news articles because it is easily accessible, subsidized and readily available with one click. However, simultaneously, it also helps to spread false news that has significant negative effects on society, that is, messages that are deliberately misinformed.

- collect Detecting fake news is a layered process that involves analysis of the news contents to determine the truthfulness of the news. The news could contain information in various formats such as text, video, image, etc. Combinations of different types of data make the detection process difficult. In addition, raw data ted is always expected to be unstructured and contain missing values in the data

- Given that online data are likely to have a considerable portion of missing values and be able to provide different forms of auxiliary information useful for creating alternative imputed values, multiple imputations are of great importance for fake news research.

- There have been many methods that are proposed ranging from NLP analysis to clustering. Amer applied two machine learning supervised algorithms, i.e., Random Forest and decision tree

DOCUMENTS:

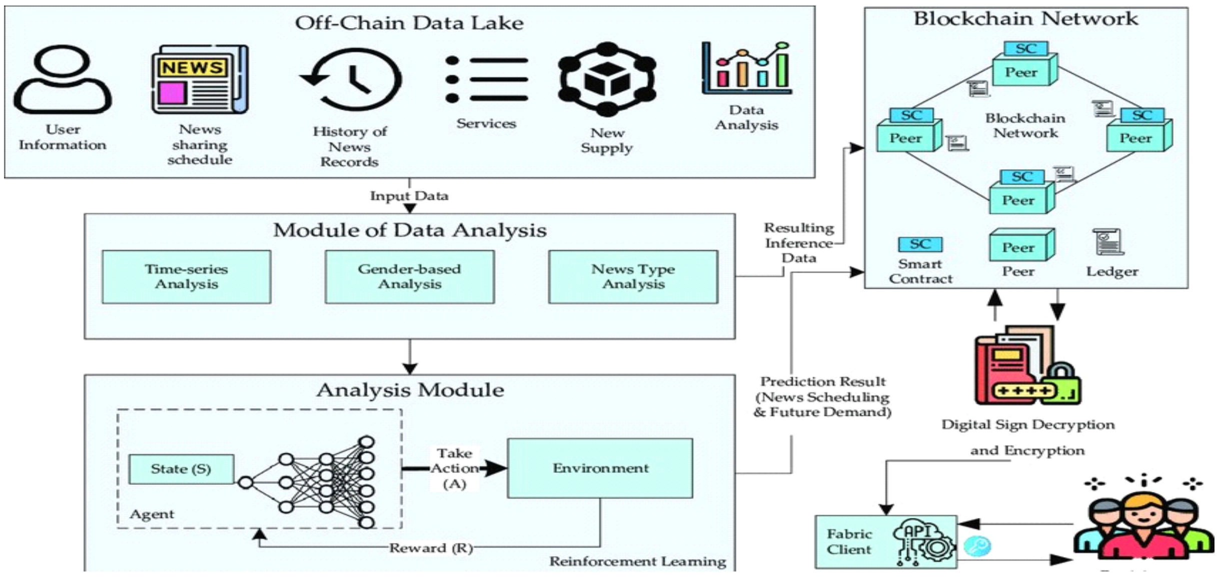
1. PROMBLEM STATEMENTS:

- Therefore, fake news detection on social media has recently become emerging research that is attracting tremendous attention.

- Fake news detection on social media presents unique characteristics and challenges that make existing detection algorithms from traditional news media ineffective or not applicable.

- 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; therefore, we need to include auxiliary information, such as user social engagements on social media, to help decide.

- Second, exploiting this auxiliary information is challenging in and of itself as users' social engagements with fake news produce data that is big, incomplete, unstructured, and noisy



2.DESIGN THINKING :

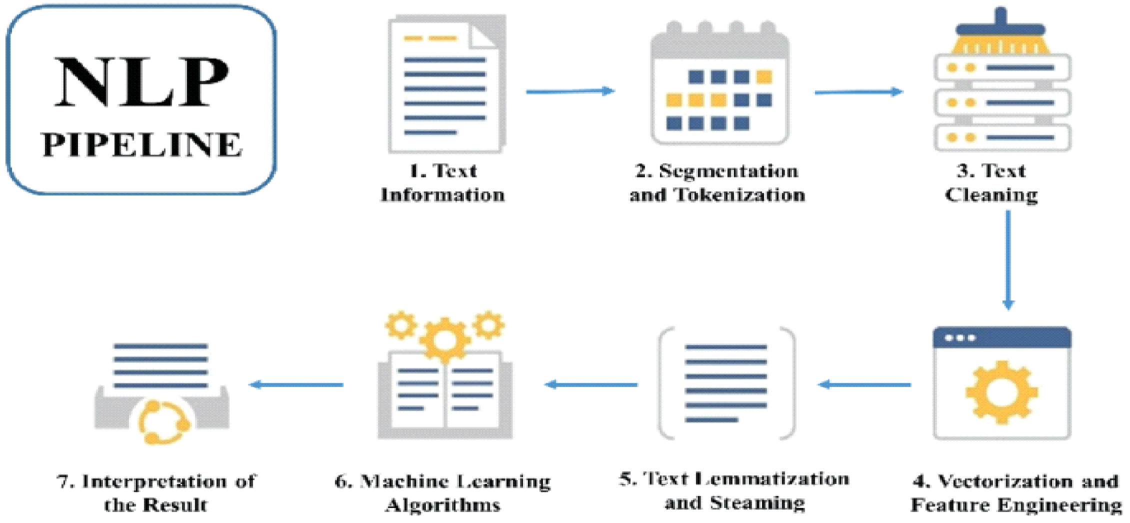
- ***DATA SOURCE:*** *Choose the fake news dataset available on Kaggle containing articles tittles and text along with their labels (genuine or fake)*

- ***DATA PREPROCESSING:*** *Clear and preprocess the textual data to prepare it for analysis.*

- ***FEATURE EXTRACTION:*** *Utilize techniques life tf-1DF (term frequency – inverse document frequency) or word embeddings to convert text into numerical features.*

- ***MODEL SELECTION :*** *Select a suitable classification algorthim (eg.logistic regression, random forest or nerual networks) for the fake news detection task.*

- ***MODEL TRAINING :*** *Train the selected model using the preprocessing data.*


NLP
PIPELINE
1. Text Information
2. Segmentation and Tokenization
3. Text Cleaning
4. Vectorization and Feature Engineering
5. Text Lemmatization and Steaming
6. Machine Learning Algorithms
7. Interpretation of the Result

3. *Phase Development:*

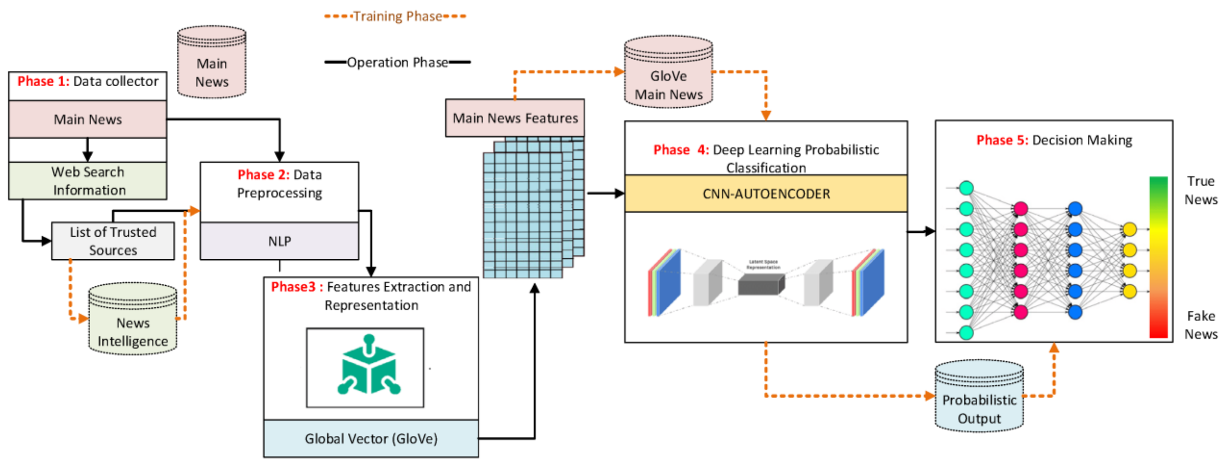
- ***Data Collection:*** Gather a diverse and extensive dataset of news articles, both real and fake, for training and testing your detection model.

- ***Data Preprocessing***: Clean and preprocess the data, including text normalization, tokenization, and removing irrelevant information.

- ***Testing:*** Test the model on a separate, unseen dataset to assess its generalization capabilities.

- ***Deployment:*** If the model performs well, deploy it for real-time or batch processing of news articles to detect fake news.

- ***Ethical Considerations:*** Ensure the responsible and ethical use of the model, considering potential biases and the impact on free speech



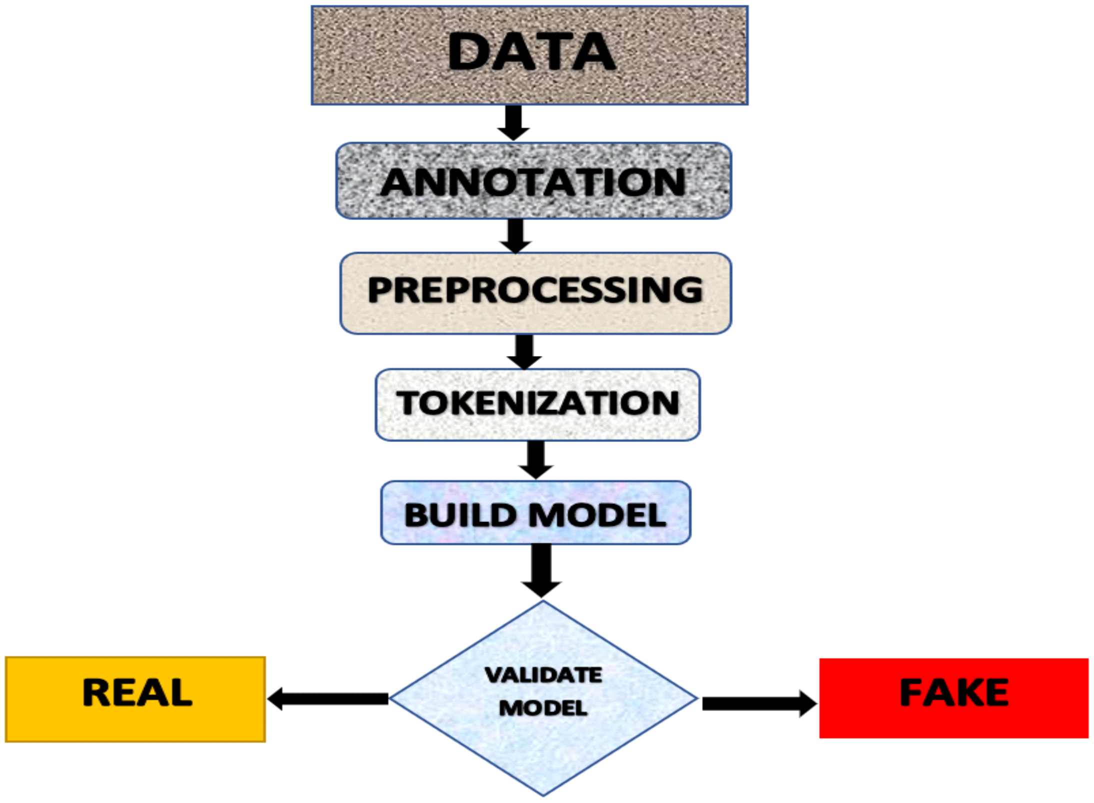
4. . Dataset Preprocessing:

- ***Text Cleaning:*** *Remove HTML tags, if any, from the text.Remove special characters and punctuation.Convert text to lowercase for consistency .*

- ***Tokenization:****Split text into words or tokens. This is important for NLP task*

- **Text Vectorization**:Convert text data into numerical vectors using techniques like TF-IDF or Word Embeddings (Word2Vec, GloVe, etc.)

- **Labels:**Encode the labels as binary values (0 for real news, 1 for fake news) or using one-hot encoding.Save Preprocessed.



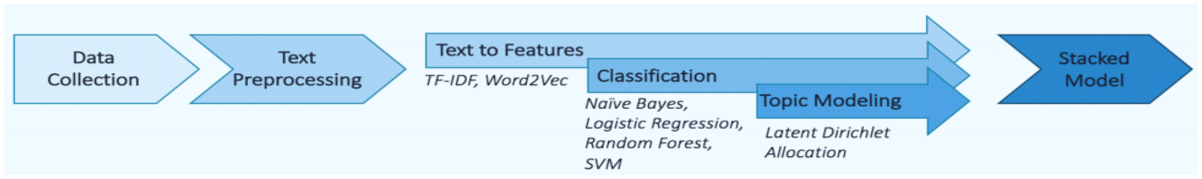
5.DATASET USED:

- The dataset contains two types of articles fake and real News. This dataset was collected from realworld sources; the truthful articles were obtained by crawling articles from Reuters.com (News website). As for the fake news articles, they were collected from different sources.

- The fake news articles were collected from unreliable websites that were flagged by PolitiFact (a fact-checking organization in the USA) and Wikipedia. The dataset contains different types of articles on different topics; however, the majority of articles focus on political and World news topics.

- The dataset consists of two CSV files. The first file named “True.csv” contains more than 12,600 articles from reuter.com. The second file named “Fake.csv” contains more than 12,600 articles from different fake news outlet resources. Each article contains the following information:

- The data collected were cleaned and processed, however, the punctuations and mistakes that existed in the fake news were kept in the text. The following table gives a breakdown of the categories and number of articles per category.



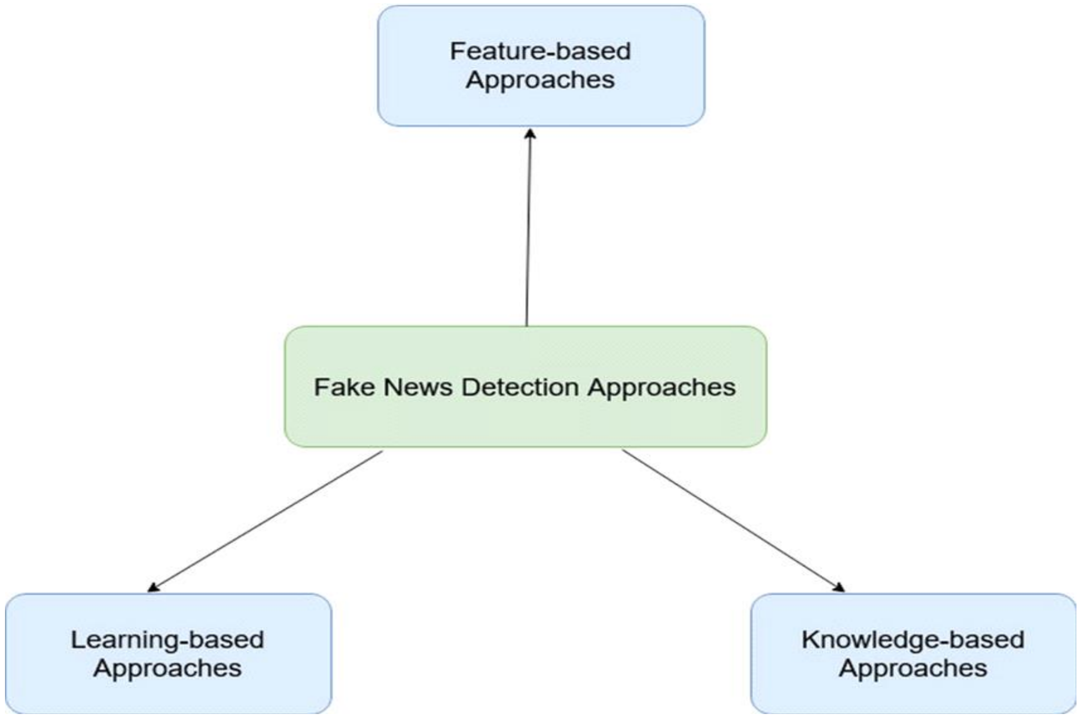
6. Feature Exaction Techniques:

- ***TF-IDF (Term Frequency-Inverse Document Frequency):*** This technique measures the importance of words in a document relative to a collection of documents. It can help identify significant terms in both real and fake news articles.

- ***Word Embeddings***: Word embeddings, such as Word2Vec or GloVe, can convert words into dense vectors, capturing semantic relationships. These vectors can be used as features for detecting fake news.

- ***N-grams***: N-grams represent sequences of words, and they can capture the linguistic patterns often found in fake news. By analysing n-grams, you can identify common phrases or combinations of words.

- ***Stance Analysis***: Analyzing the stance of the news article towards a particular topic or entity can help detect biased or fake news.


Feature-based Approaches
Fake News Detection Approaches
Learning-based Approaches
Knowledge-based Approaches

7. Future Extraction Algorithm:

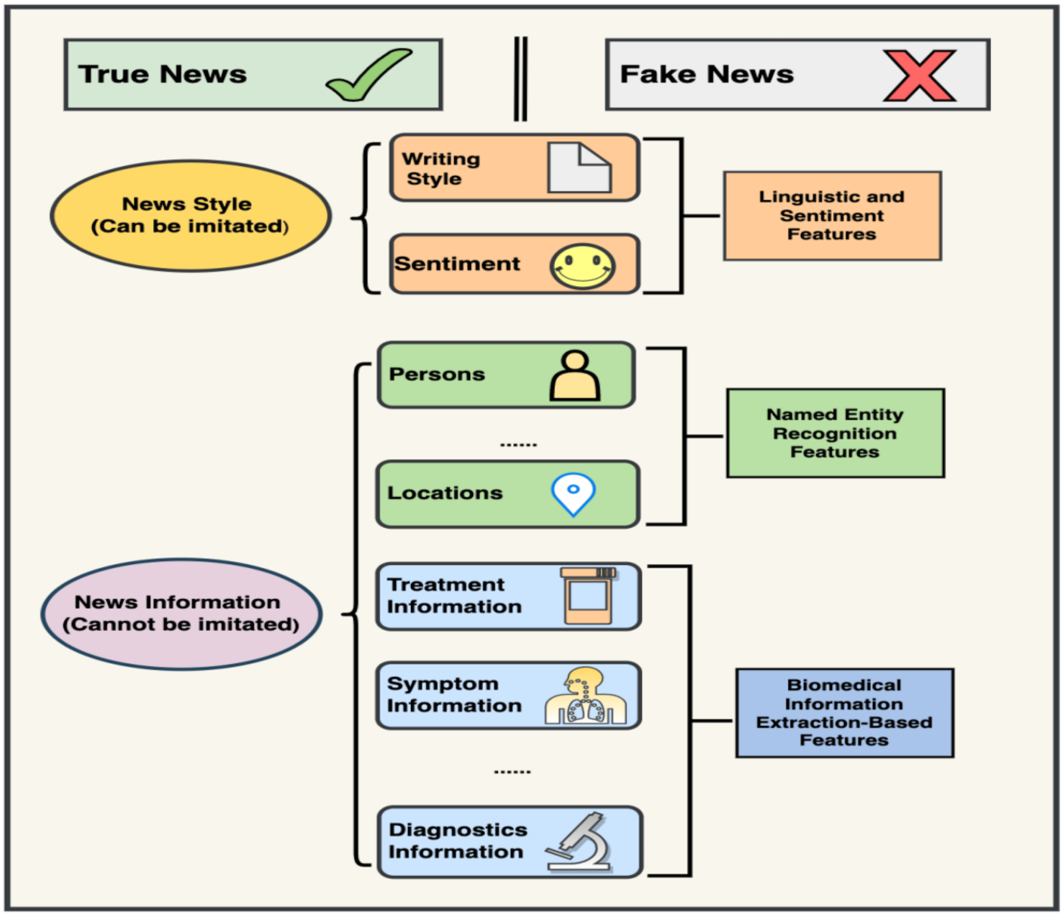
- ***TF-IDF (Term Frequency-Inverse Document Frequency):*** Measure the importance of words in the document relative to a corpus.

- ***N-grams:*** Extract sequences of N consecutive words (e.g., bigrams, trigrams)

- ***Syntax Features***: Extract grammatical features, such as part-of-speech tags.

- ***Multimodal Features***:If available, consider features from images or videos associated with the news.

- ***Contextual Features***:Analyze the context in which the news is presented and how it aligns with known for event


True News
Fake News
News Style (Can be imitated)
Writing Style
Sentiment
Linguistic and Sentiment Features
Persons
......
Locations
Named Entity Recognition Features
News Information (Cannot be imitated)
Treatment Information
Symptom Information
......
Diagnostics Information
Biomedical Information Extraction-Based Features

8. Model Training Processing:

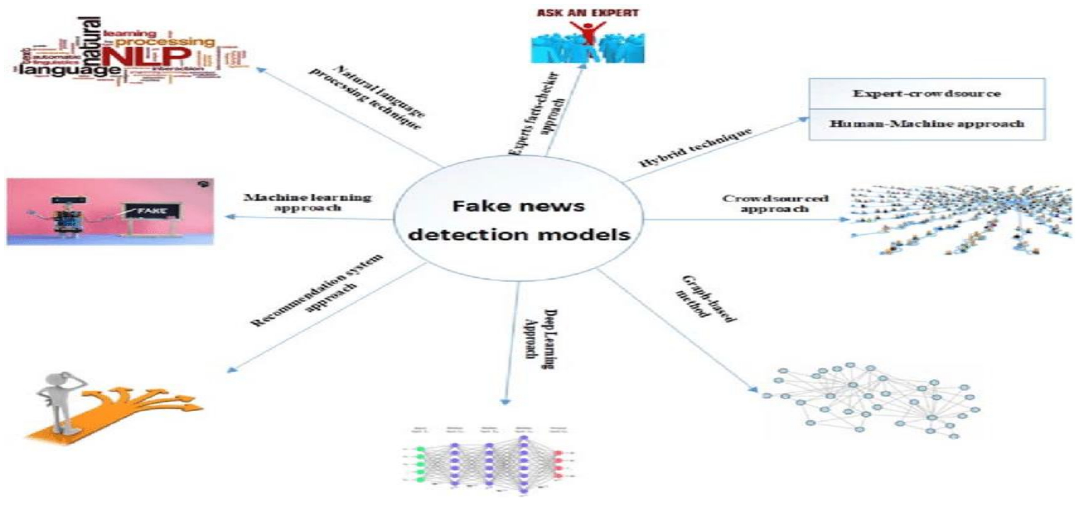
- ***Needs Assessment*****:** The first step in the training process is to assess the need for training the employees. It analyses what are the long-term requirements of the organization and what does the organization expect from the employees.

- ***Defining Training Objective***: After deriving the learning gap, organizations should define the learning objective. Goals and objectives of training become the foundation of the training initiatives. Hence determining the training objectives gives direction to the entire learning program.

- ***Designing a Training Program:*** Once the objective of the training program is determined, it is time to analyse the factors that need to be considered while designing a training program.

- ***Implementation of the Training Program***: After designing a training program, it is time to implement it.

- ***Evaluation and Follow-up***: The final step in the training process is to evaluate and follow up on the effectiveness of the training program.



SUBMISSION:

- ***Program:***

```
import pandas as pd
import matplotlib.pyplot as plt
import spacy
 from spacy.util import minibatch, compounding
import random
 nlp = spacy.load('el__core__news__md')
 df1 = pd.read__csv('../data/jtp__fake__news.csv')
 df1.replace(to__replace='[ \ n \ r \ t]', value=' ', regex=True, inplace=True)
 def load__data(train__data, limit=0, split=0.8):
 random.shuffle(train__data)
 train__data = train__data[-limit:]
 texts, labels = zip(*train__data)
```

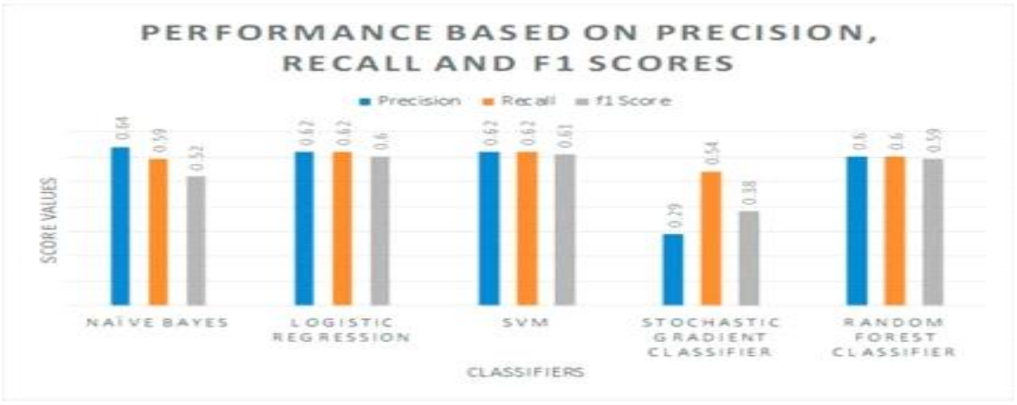
```
cats = [{"REAL": not bool(y), "FAKE": bool(y)} for y in l abels]
 split = int(len(train__data) * split)
return (texts[:split], cats[:split]), (texts[split:], cats[split:])
 # - - - - - - - - - - - - - - - - - - evaluate function defined
                                                below- - - - - - - - - - - -
 def evaluate(tokenizer, textcat, texts, cats):
 docs = (tokenizer(text) for text in texts)
tp = 0.0 # True positives
 fp = 1e-8 # False positives
 fn = 1e-8 # False negatives
 tn = 0.0 # True negatives
 for i, doc in enumerate(textcat.pipe(docs)):
 gold = cats[i]
for the label, score in doc.cats.items():
 if the label is not in gold:
 Continue
 if label = = "FAKE":
 Continue
 if score > = 0.5 and gold[label] > = 0.5:
 tp + = 1.0
 elif score > = 0.5 and gold[label] < 0.5:
 fp + = 1.0
 elif score < 0.5 and gold[label] < 0.5:
 tn + = 1
elif score < 0.5 and gold[label] > = 0.5:
 fn + = 1
 precision = tp / (tp + fp)
recall = tp / (tp + fn)
 #- - - - - - - - - - - -if conditions for precision recall - - - - - - - - -
 if (precision + recall) = = 0:
f__score = 0.0
 else:
 f__score = 2 * (precision * recall) / (precision + recall)
 return {"textcat__p": precision, "textcat__r": recall,
"textcat__f": f__score}
 In [3]:
 df1.info()
 RangeIndex: 100 entries, 0 to 99
Data columns (total five columns):
 # Column Non-Null Count Dtype
-- - - - - - - - - - - - - - - - - - - - - - - - - -
 0 title 100 non-null object
 One text 100 non-null object
 Two sources 100 non-null object
 Three url 100 non-null object
 4 is__fake 100 non-null int64
dtypes: int64(1), object(4)
memory usage: 4.0+ KB
 textcat=nlp.create__pipe( "textcat",
 config={"exclusive__classes": True, "architecture":
"simple__cnn"})
 nlp.add__pipe(textcat, last=True)
 nlp.pipe__names
 ['tagger', 'parser', 'ner', 'textcat']
 textcat.add__label("REAL")
 textcat.add__label("FAKE")
 df1['tuples'] = df1.apply(lambda row: (row['text'],
 row['is__fake']), axis=1)
 train = df1['tuples'].tolist()
 (train__texts, train__cats), (dev__texts, dev__cats) =
 load__data(train, split=0.9)
 train__data = list (zip(train__texts,[{'cats': cats} for cats in
 train__cats]))
 n__tier = 20
 # - - - - - - - - - - - - Disabling other components- - - - - - - - - - - - -
 other__pipes = [pipe for pipe in nlp.pipe__names if pipe!=
 'text cat']
 with nlp.disable__pipes(*other__pipes): # only train
 Text cat
 optimizer = nlp.begin__training()
print ("Training the model...")
 print ('{: ^5} \t {: ^5} \t {: ^5} \t {: ^5}'. format ('LOSS', 'P', 'R', 'F'))
```

OUTPUT:

```
    array([1716, 1722, 122, 363, 311, 322, 236, 228, 220, 226, 223, 220, 206,
202, 283, 282, 280, 278, 275, 266, 266, 261, 262, 256, 255, 253, 252, 215, 211,
213, 237, 233, 232, 232, 230, 226, 228, 225, 221, 223, 222, 222, 220, 226, 228,
227, 226, 221, 222, 220, 206, 208, 206, 205, 201, 203, 202, 202, 200, 66, 68,
67, 66, 65, 61, 63, 62, 60, 86, 88, 87, 86, 81, 83, 82, 76, 78, 77, 76, 75, 71, 73,
72, 72, 70, 66, 68, 67, 66, 65, 61, 63, 62, 62, 60, 56, 58, 57, 56, 55, 51, 53, 52,
52, 50, 16, 18, 17, 16, 15, 11, 13, 12, 12, 10, 36, 38, 37, 36, 35, 31, 33, 32, 32, 30,
```

26, 28, 27, 26, 25, 21, 23, 22, 221, 223, 222, 222, 220, 226, 228, 227, 226, 221, 222, 220, 206, 208, , 280, 278, 275, 266, 266, 261, 262, 256, 255, 253, 252, 215, 211, 213, 237, 233, 232, 232, 230, 226, 228, 225, 221, 223, 222, 222, 220, 226, 228, 227, 226, 221, 222, 206, 205, 201, 203, 202, 202, 200, 66, 68, 67, 66, 65, 61, 63, 62, 60, 86, 88, 87, 86, 81, 83, 82, 76, 78, 77, 76, 22, 20, 26, 28, 27, 26, 25, 21, 23, 22, 22, 20, 6, 8, 7, 6, 5, 1, 3, 2, 2])

PERFORMANCE GRAPHS OF CLASSIFIERS:



Conclusion:

- The study of fake news detection was carried out using the ISOT and LIAR datasets with real and fake news contents from Reuters.com, PolitiFact and Fake Newsnet.

- Initially, the proposed technique selects important feature terms relying on the parts of speech (POS) in the textual information, and then uses sentiment analysis to estimate users' control variables for opinions using lexicon-based scoring analysis.

- For improving classification-based false news identification, a data imputation preparation approach is presented. This approach is based on the utilisation of data imputation techniques to handle missing values in a dataset.

- Subsequently, the term frequency and inverse document frequency (TF-IDF) were used for the extraction of useful features from the datasets to help the detection accuracy.

- Finally, the fake news was detected using multiple classification models. Initially, for the multiclass prediction and robustness of predicting the class of text, the Naïve Bayes model had been used. Secondly, the passive-aggressive classifier trains the model incrementally and eventually, the deep neural network was used to increase the efficiency to detect fake news.