FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME ,N-GRAM ANALYSIS,DECISION TREE MACHINE LEARNING TECHNIQUE

A PROJECT REPORT

submitted

in the partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

R.KEERTHI-17B81A0592

YAKKATI KAILASH-17B81A0583

MADAVARAM MANIDEEP-17B81A05B2

Under the guidance of

MS.G.RAMYA

ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CVR COLLEGE OF ENGINEERING

(*An Autonomous institution, NBA, NAAC Accredited and Affiliated to JNTUH, Hyderabad*)

Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Rangareddy (D), Telangana- 501 510

May 2021

CERTIFICATE

This is to certify that the project entitled **“FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE”** being submitted by **R.KEERTHI 17B81A0592, YAKKATI KAILASH 17B81A0583, MADAVARAM MANIDEEP 17B81A05B2** in partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering to the CVR College ofEngineering , is a record of bonafide work carried out by them under my guidance and supervision during the year 2020-2021.

The results embodied in this project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Signature of the project guide	Signature of the HOD
Ms.G.Ramya	**Dr. A. Vani Vathsala**
Assistant Professor	

External Examiner

ACKNOWLDEGEMENT

We owe our gratitude to **Dr. Raghava V Cherabuddi,** Chairman, CVR College of Engineering, Ibrahimpatnam, for extending the college facilities to the successful pursuit of our project.

We wish to record our profound gratitude to **Dr. Nayanthara K Sattiraju,** the Principal of CVR College of Engineering, for motivation and encouragement,

We are indebted to **Dr A. Vani Vathsala,** Head of the Department, for the permission to carry out this project.

We owe a great thanks to our project coordinator **Dr. N. Subhash Chandra,** for supporting us throughout the project.

It is our pleasure to express our heartfelt thanks to **Ms.G.Ramya,** Assistant Professor Department of Computer Science and Engineering, for her supervision and guidance which enabled us to understand and develop the project.

We owe sincere thanks to all the faculty members in the department of Computer Science & Engineering for their kind guidance and encouragement from time to time.

We take this opportunity to offer our regards to all of those who have supported us directly or indirectly in the successful completion of this project work.

We acknowledge our deep sense of gratitude to our parents for being a constant source of inspiration and motivation.

R.KEERTHI - 17B81A0592

YAKKATI KAILASH - 17B81A0582

M.MANIDEEP - 17B81A05B2

ABSTRACT

In this contemporary world, using social networks like Facebook or Twitter in order to find news is something usual. Users from these networks create a lot of content, not only posts, likes, comments but also redistribution of information with retweet option or with simple copy/paste operations. A common problem here is the detection of fake news and it is often difficult to distinguish the creditability of news that appears that appears in a social networking.

Fake news is news articles providing wrong information to readers. Normally these articles are originated from social networking sites which immensely influence young generation as well as various sectors of life. Among the few biggest sources of rumors, fake news and fake information are social networking sites like Facebook and Twitter. Detection of fake news on Twitter is challenging as the amount of resources is limited; less datasets are available, and also a large amount of data is generated by these social networking sites. It is very difficult to predict if the data is fake or not even after having knowledge in the same area. In this project, we would propose a method for detection of fake news on Twitter which involves collecting data of Twitter using flumes and performing *N*-gram analysis on it for feature extraction and then applying decision tree machine learning technique to classify documents as fake or real.

TABLE OF CONTENTS

List of Figures

1. INTRODUCTION

1.1 MOTIVATION

Twitter has proved to be very useful because of its ability to propagate news much faster than traditional media. News on Twitter can come from authorized news organizations, but most of them come from public users. Unlike traditional media sources, the absence of supervision and the ease of spreading make Twitter an environment conducive to rumors and fake news . This issue becomes a problem as more people rely on social media for news especially during emergencies. Hence, determining the credibility of the content on Twitter is highlighted especially in the case of emergencies.

1.2 PROBLEM STATEMENT

In the recent years, online content has been playing a significant role in swaying users decisions and opinions. Opinions such as online reviews are the main source of information for e-commerce customers to help with gaining insight into the products they are planning to buy. Even though the problem of fake news is not a new issue, detecting fake news is believed to be a complex task given that humans tend to believe misleading information and the lack of control of the spread of fake content.

This project focuses on the credibility problem and introduces a supervised learning model based on word N-gram analysis and machine learning techniques to automatically classify tweets into credible and not credible.

1.3 OBJECTIVES OF THE PROJECT

News on Twitter can come from authorized news organizations, but most of them come from public users. Unlike traditional media sources, the absence of supervision and the ease of spreading make Twitter an environment conducive to rumors and fake news. Several studies have shown that much of the content on Twitter is not credible.

Classification based approaches classify tweets into credible and not credible based on features extracted from them using machine learning techniques especially supervised techniques. Supervised machine learning techniques require a ground truth that contains a dataset of annotated tweets with the features related to them. The relevance of the extracted features is an important factor affecting the efficiency of the prediction. After the feature dataset is built, the next step is to determine the optimal classification algorithm to train them. Decision trees and support vector machines are the most popular supervised learning techniques used for classification. To predict the credibility of a tweet, we should consider the content of the tweet as an important factor. This project focuses on the credibility problem and introduces a supervised learning model based on word N-gram analysis and machine learning techniques to automatically classify tweets into credible and not credible.

1.4 PROJECT REPORT ORGANIZATION

- Chapter 2 gives insights about proposed model,introduces characteristics of problem and design challenges
- Chapter 3 summarizes functional,non-functional requirements and system requirements along with software and hardware specifications.
- Chapter 4 deals with analysis and design of the proposed model
- Chapter 5 encloses Implementation of the proposed model and testing
- Chapter 6 includes conclusion and future work
- Chapter 7 includes references

2. LITEARTURE SURVEY

2.1 EXISTING SYSTEM

For our project we got motivation by the research carried out by the following people and their published papers:

[1] Castillo et al. were the first group to work on the problem of credibility and proposed a model that automatically classify tweets based on features extracted from them . The research identified different types of features. Some features are related to the content or the author of the tweet while others are aggregated from the related topic. The extracted features were used to train a set of classifiers like SVM, Bayesian networks and decision trees. They achieved credibility classification with an accuracy of nearly 86% using the J48 decision tree. The research provided a feature analysis to perform the best feature selection process. The study indicated that the best features are related to the users such as the duration they spent as Twitter users, the number of followers that they have, and the number of tweets that they have written.

[2] Gupta et al. proved that predicting the credibility of Twitter messages could be automated accurately. The research identified some relevant features such as the number of followers, number of unique characters and swear words. Results showed that approximately 30% of tweets posted in an event include information about the event, 14% of the tweets related to an event were spam while only 17% are credible tweets.

[3] Zubiaga et al. developed a credibility detection system that warns users of unverified posts. Twitter streaming API was used to collect 5802 tweets related to five breaking news stories. The research evaluated and compared the performance of the system using two different feature sets: content-based features and social features. Conditional Random Fields (CRF) was used as a sequential classifier and its performance was compared with three more classifiers. The experimental results showed that the features related to the text of the tweet itself (such as word vectors, word count and the existence of question marks) are good indicators for credibility.

[4] Al-Khalifa et al. developed a model to measure credibility of Twitter messages and assign the credibility level (high, low and moderate) to each tweet. The proposed model is based on the similarity between Twitter messages and authorized news sources like Aljazeera.net. The proposed model achieved acceptable results but requires the existence of credible external sources.

[5] "Twitter by the numbers: Stats, Demographic & Fun Facts." [Online]. Available: https://www.omnicoreagency.com/twitterstatistics/ [Accessed: 17-10-2019]

In recent years, research on Twitter sentiment analysis, which analyzes Twitter data (tweets) to extract user sentiments about a topic, has grown rapidly. Many researchers prefer the use of machine learning algorithms for such analysis. This study aims to perform a detailed sentiment analysis of tweets based on ordinal regression using machine learning techniques. The proposed approach consists of first pre-processing tweets and using a feature extraction method that creates an efficient feature. Then, under several classes, these features scoring and balancing. Multinomial logistic regression (SoftMax), Support Vector Regression (SVR), Decision Trees (DTs), and Random Forest (RF) algorithms are used for sentiment analysis classification in the proposed framework. For the actual implementation of this system, a twitter dataset publicly made available by the NLTK corpora resources is used. Experimental findings reveal that the proposed approach can detect ordinal regression using machine learning methods with good accuracy. Moreover, results indicate that Decision Trees obtains the best results outperforming all the other algorithms.

[6] A. Stocker, A. Richter, and K. Riemer. "A Review of Microblogging in the Enterprise", itInformation Technology Methoden und innovative Anwendungen der Informatik und Informationstechnik Vol. 54, No. 5, pp. 205-211, 2012.

Introduction This chapter examines some reasons why project teams would use social media. Social media were just beginning to be co-opted by business interests in the 2010s (Kiron, Palmer, Phillips, & Kruschwitz, 2012): early adopting firms – the ones that tend to adopt social media applications – can be viewed as innovators, driven by efficiency and profit gains (e.g., Delerue & Cronje, 2015; Perrigot, Kacker, Basset, & Cliquet, 2012). Social media generally provide: Web-based platforms that allow workers to (1) communicate messages with specific coworkers or broadcast messages to everyone in the organization, (2) articulate a list of coworkers with whom they share a connection, (3) post, edit, and sort text and files linked to themselves or others, and (4) view the messages, connections, text, and files communicated, articulated, posted, edited and sorted by anyone else in the organization at any time of their choosing. (Leonardi, Huysman, & Steinfield, 2013, p. 2) The development of complex products, services, and processes with very short time-to-market combined with needs for cross-functional expertise have compelled increasing numbers of organizations to implement their business operations as projects (Kerzner, 2002). Projects have been described as temporary organizations that are strongly focused on a defined task, and therefore very agile. In addition, projects are ephemeral in the sense that the knowledge that is gained through the project quickly evaporates after the project team is disbanded (Gemünden, 2013, p. 2). An essential component of project execution is the information that feeds decision making and knowledge creation. Some researchers therefore view social media – which are used to convey information – as valuable for project management (Harrin, 2010; Remidez & Jones, 2012; Rimkuniene & Zinkeviciute, 2014). For instance, Rimkuniene and Zinkeviciute (2014) demonstrate how social media can enhance effective communication in temporary organizations by addressing specific project-based needs. Remidez and Jones (2012) emphasize that project managers must understand the relationships between communication practices and trust development, and how they are affected by social media. According to Harrin's (2010) study on social media in project environments, over two-thirds of 181 project managers surveyed across thirty-two countries believed that social media constitute a key issue for their industry (Harrin, 2010).

[7]] X. Lu, and C. Brelsford. "Network structure and community evolution on twitter: human behavior change in response to the 2011 Japanese earthquake and tsunami.", Scientific reports 4, p. 6773, 2014.

To investigate the dynamics of social networks and the formation and evolution of online communities in response to extreme events, we collected three datasets from Twitter shortly before and after the 2011 earthquake and tsunami in Japan. We find that while almost all users increased their online activity after the earthquake, Japanese speakers, who are assumed to be more directly affected by the event, expanded the network of people they interact with to a much higher degree than English speakers or the global average. By investigating the evolution of communities, we find that the behavior of joining or quitting a community is far from random: users tend to stay in their current status and are less likely to join new communities from solitary or shift to other communities from their current community. While non-Japanese speakers did not change their conversation topics significantly after the earthquake, nearly all Japanese users changed their conversations to earthquake-related content. This study builds a systematic framework for investigating human behaviors under extreme events with online social network data and our findings on the dynamics of networks and communities may provide useful insight for understanding how patterns of social interaction are influenced .

2.2 LIMITATIONS OF EXISTING SYSTEM

From the above studied papers, we have observed some of the limitations which we tried to overcome in our project.
The limitations are as follows:

- Serious fabrications are news not published in mainstream or participant media, yellow press or tabloids, which as such, will be harder to collect.
- Large-Scale hoaxes are creative and unique and often appear on multiple platforms.
- The authors argued that it may require methods beyond text analytics to detect this type of fake news.
- Humorous fake news, are intended by their writers to be entertaining, mocking, and even absurd According to the authors, the nature of the style of this type of fake news could have an adverse effect on the effectiveness of text classification techniques.

2.3 PROPOSED SYSTEM

We aim to develop a model for automatically classifying tweets into credible and not credible. In this section, we will discuss our model which is based on text analysis using word N-grams. The annotated dataset is input to the preprocessing step then N-gram features are extracted in order to create feature vectors. The feature vectors with their credible/not-credible labels are passed to the classifier training process to learn different tweet patterns and minimize the classification error.

2.4 MERITS OF PROPOSED SYSTEM

- N-gram modeling is a popular feature identification and analysis approach used in language modeling and Natural language processing fields.
- N-gram is a contiguous sequence of items with length n. It could be a sequence of words,

bytes, syllables, or characters. The most used n-gram models in text categorization are word-based and character-based n-grams. In this work, we use word-based n-gram to represent the context of the document and generate features to classify the document.

- We develop a simple n-gram based classifier to differentiate between fake and honest news articles. The idea is to generate various sets of n-gram frequency profiles from the training data to represent fake and truthful news articles. We used several baseline n-gram features based on words and examined the effect of the n-gram length on the accuracy of different classification algorithms.

3. REQUIREMENT SPECIFICATION

3.1 SOFTWARE REQUIREMENTS

The Software requirements of our project are as follows:

• Operating system	:	Windows 7,8,10 Ultimate, Linux, Mac.
• Front-End	:	Python.
• Coding Language	:	Python.
• Software Environment	:	Anaconda(jupyter or spyder).

3.1.1 PYTHON3:

Python is a high level, interpreted and general purpose dynamic programming language. It supports several technologies like machine learning, database management etc. Python has several predefined packages or modules which can be used easily for solving problems.

Python is widely used in the software development industry. There are many of reasons for this.

i. **High-level object-oriented programming language**: Python includes effective symbolism.

ii. **Rapid application development**: Because of its concise code and literal syntax, the development of applications gets accelerated. The reason for its wide usability is its simple and easy-to-master syntax. The simplicity of the code helps reduce the time and cost of development.

iii. **Dynamic typescript**: Python has high-level incorporated data structures blended with dynamic typescript and powerful binding.

Some of the unique features that make Python the most ubiquitous language among the developer community are:

a) Python supports code reusability and modularity.

b) It has a quick edit-inspect-debug cycle.

c) Debugging is straightforward in Python programs.

d) It has its own debugger written in Python itself, declaring to Python’s reflective power. e) Python includes a plethora of third-party components present in the Python Package Index (PyPI).

3.1.2 MODULES:

1. **Numpy**

Python has a strong set of data types and data structures. Numpy is a data handling library, particularly one which allows us to handle large multi-dimensional arrays along with a huge collection of mathematical operations. Numpy isn’t just a data handling library known for its capability to handle multidimensional data. It is also known for its speed of execution and vectorization capabilities. It provides MATLAB style functionality and hence requires some learning before you can get comfortable.It is also a core dependency for other majorly used libraries like pandas, matplotlib and so on.It’s documentation itself is a good starting point.

2. **Pandas**

Pandas is a python library that provides flexible and expressive data structures (like dataframes and series) for data manipulation. Built on top of numpy, pandas is as fast and yet easier to use. Pandas provides capabilities to read and write data from different sources like CSVs, Excel, SQL Databases, HDFS and many more. It provides functionality to add, update and delete columns, combine or split dataframes/series, handle datetime objects, impute null/missing values, handle time series data, conversion to and from numpy objects and so on.

3.Scipy

Pronounced as Sigh-Pie, this is one of the most important python libraries of all time. Scipy isa scientific computing library for python. This is yet another *behind the scenes* library which does a whole lot of heavy lifting. It providesmodules/algorithms for linear algebra, integration, image processing, optimizations, clustering, sparse matrix manipulation and many more.

4. Matplotlib

Another component of the SciPy stack, matplotlib is essentially a visualization library. It worksseamlessly with numpy objects (and its high-level derivatives like pandas). Matplotlib providesa MATLAB like plotting environment to prepare high-quality figures/charts for publications, notebooks, web applications and so on.

5. Scikit-Learn

Designed as an extension to the SciPy library, scikit-learn has become the de-facto standard for many of the machine learning tasks. Scikit-learn provides a simple yet powerful fit-transform and predict paradigm to learn from data, transform the data and finally predict. Using this interface, it provides capabilities to prepare classification, regression, clustering and ensemble models. It also provides a multitudeof utilities for preprocessing, metrics, model evaluation techniques, etc.

6. Seaborn

Seaborn is a high-level visualization library. It provides sophisticatedstyles straight out of the box. Apart from styling prowess and sophisticated color pallets, seaborn provides a range of visualizations and capabilities to work with multivariate analysis. It provides capabilities to perform regression analysis, handling of categorical variables and aggregate statistics.

3.2 HARDWARE REQUIREMENTS

The following the hardware requirements which are necessary to develop our project

• Processor	:	Intell I-3, 5, 7 Processor.
• Hard Disk	:	500 GB.
• Floppy Drive	:	1.44 Mb.
• Ram	:	2Gb.

3.3 FUNCTIONAL REQUIREMENTS

Outputs from computer systems are required primarily to communicate the results of processingto users. They are also used to provide a permanent copy of the results for later consultation.

The various types of outputs in general are:

- External Outputs, whose destination is outside the organization,.
- Internal Outputs whose destination is within organization and they are theuser’s main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.
- Understanding user’s preferences, expertise level and his business requirements througha friendly questionnaire.
- Input data can be in four different forms - Relational DB, text files, .xls and xml files. For testing and demo you can choose data from any domain. User-B can provide business dataas input.

3.4 USER REQUIREMENTS

- User has to load the application before using it
- User needs to have data (text, audio, image, video) which is to be hidden
- User needs to have a master file in which he/she wants to hide the data
- User needs to have a stego-key in order to encrypt or decrypt the data

4. DESIGN

4.1 USECASE DIAGRAM

The following use case diagram consists of one actor namely

- ✓ User

User : User interacting with the application uploads an input file. The accepted file goes through a series of steps i.e. pre-processing data, applying n gram, running Decision Tree algorithm, checking of accuracy and error rate , chec and writing into credible and fake news. Final output will be displayed to the user.

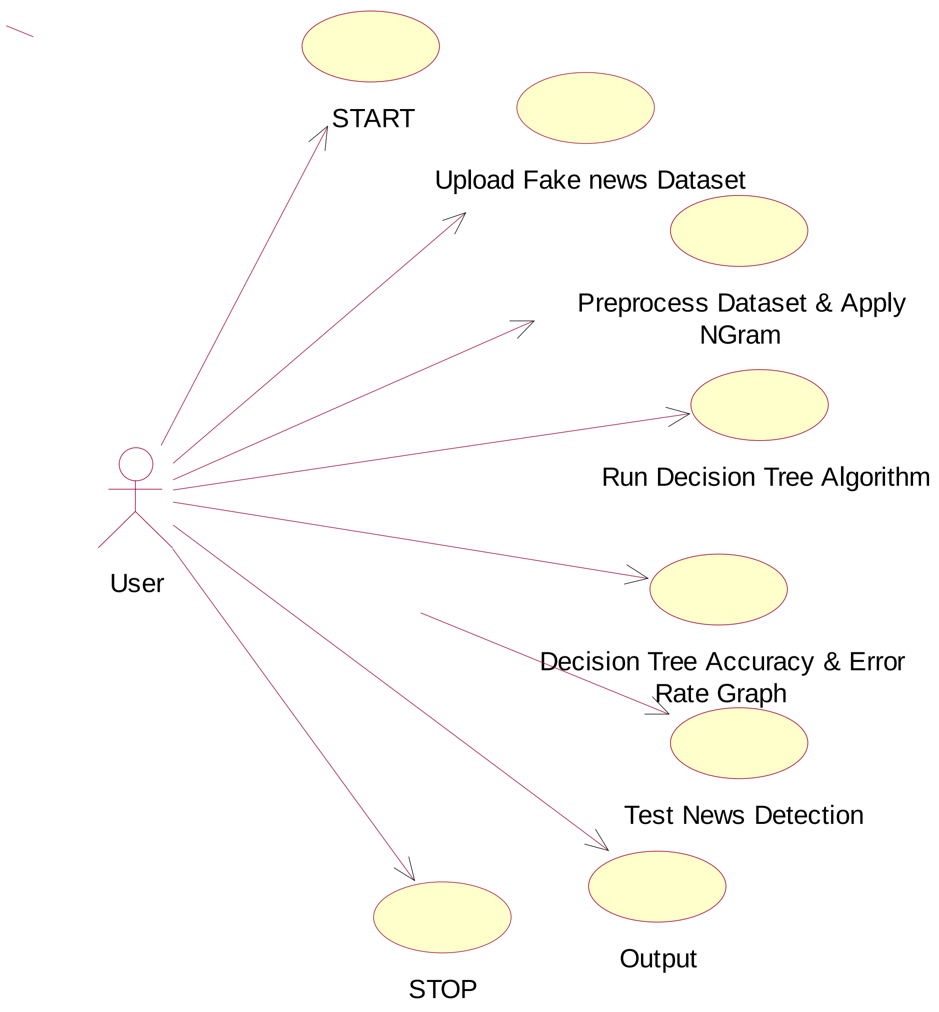

START
Upload Fake news Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
User
Decision Tree Accuracy & Error Rate Graph
Test News Detection
Output
STOP

Fig 4.1 Use case Diagram

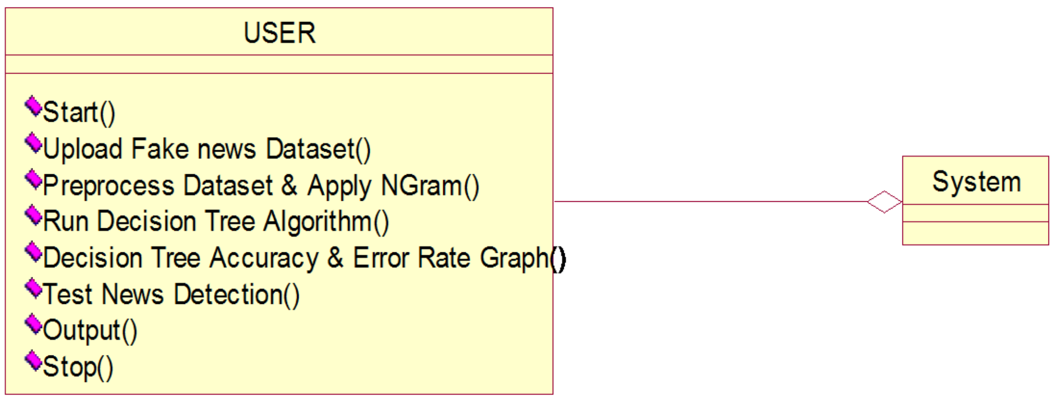
4.2 CLASS DIAGRAM

The following class diagram consists of 2 main classes.

They are

- ✓ User class
- ✓ System class

The dependencies of classes on one another and their relations along with the important variables and methods used are shown. User calls the required methods in the class while interacting with the system.



USER
Start()
Upload Fake news Dataset()
Preprocess Dataset & Apply NGram()
Run Decision Tree Algorithm()
Decision Tree Accuracy & Error Rate Graph()
Test News Detection()
Output()
Stop()
System

Fig 4.2 Class Diagram

4.3 SEQUENCE DIAGRAM

The relationship description that demonstrates how, and under what case, the procedures operate together is said to be sequence graph .

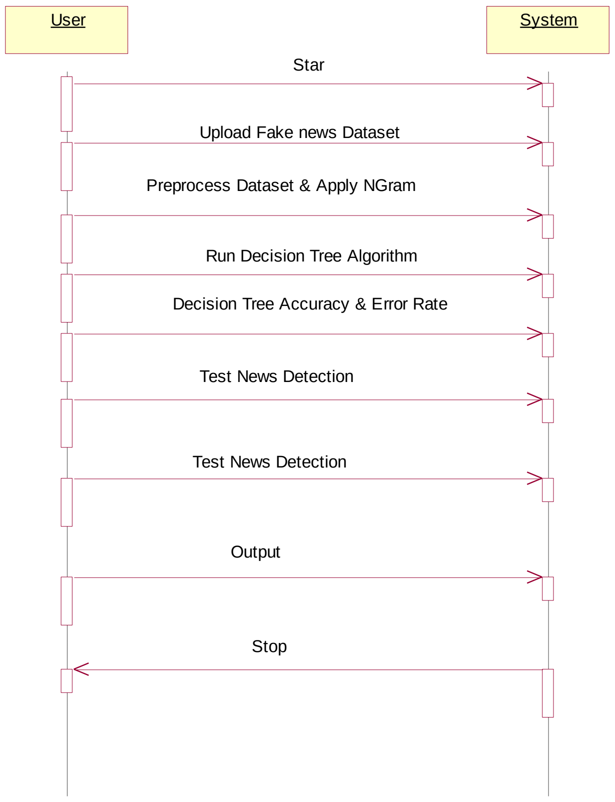

User
System
Star
Upload Fake news Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
Decision Tree Accuracy & Error Rate
Test News Detection
Test News Detection
Output
Stop

Fig 4.3 Sequence Diagram

4.4 ARCHITECTURE DIAGRAM

System architecture shows the states in the processing such as giving the processed data to algorithm analyzing for feature extraction, training the classifier and summarization whether truthful or fake and finally displaying the output. The working is explained in briefbelow:

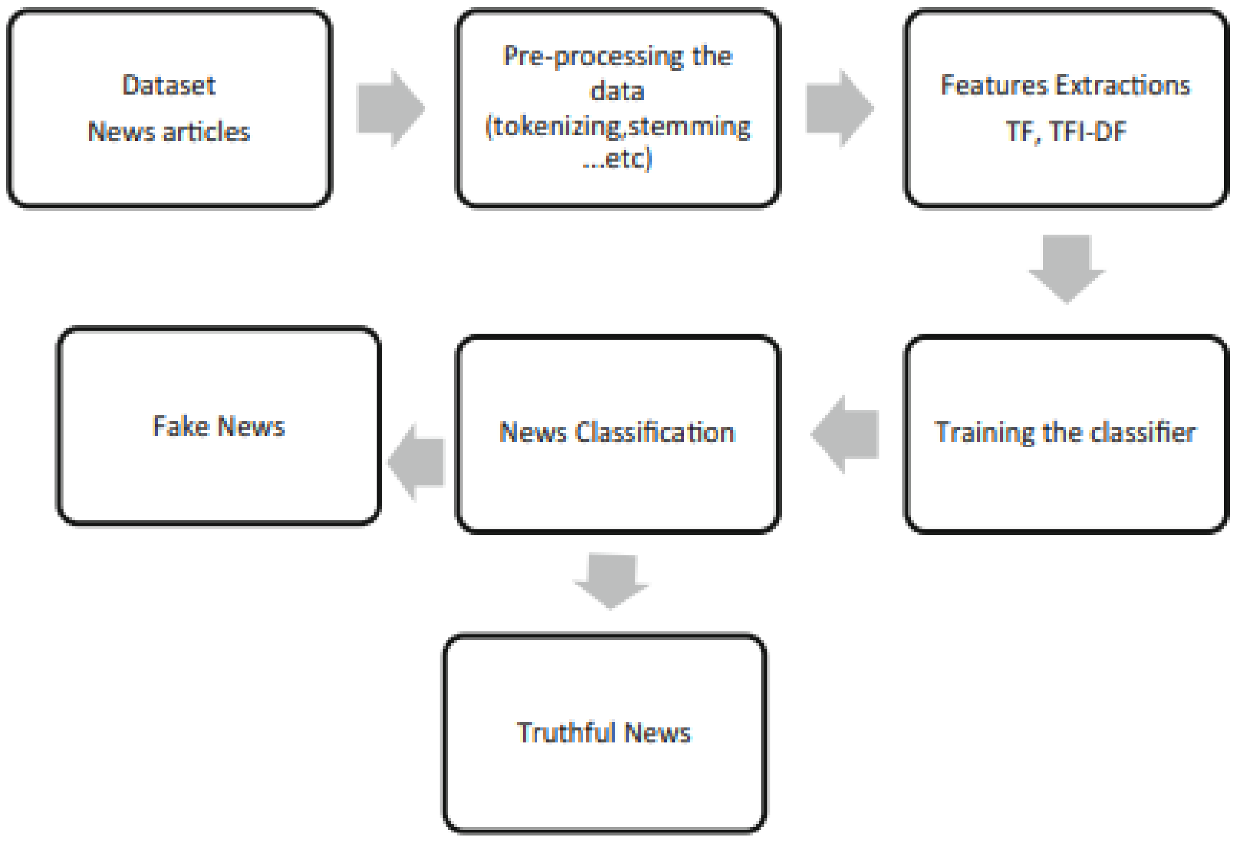


Fig 4.4 Architecture Diagram

- **DATASETS:**

  Machine learning techniques require building a dataset contains a collection of tweet messages. The messages in this dataset are then labeled by human annotators which is an important step affecting the accuracy of the model.

- **DATA PRE-PROCESSING:**

  Before representing the data using n-gram and vector-based model, the data need to be subjected to certain refinements like stop-word removal, tokenization, a lower casing, sentence

segmentation, and punctuation removal. This will help us reduce the size of actual data by removing the irrelevant information that exists in the data. We created a generic processing function to remove punctuation and non-letter characters for each document; then we lowered the letter case in the document.
TF-IDF is used in machine learning and text mining as a weighting factor for features. The weight increases as the word frequency in a document increases but that is offset by the number of times that word appears in the dataset. This mechanism helps to remove the importance from really common words that appear frequently in all documents and rewards words that are rare overall in a dataset.

- **CLASSIFICATION PROCESS:**

It starts with preprocessing the data set, by removing unnecessary characters and words from the data. N-gram features are extracted, and a features matrix is formed representing the documents involved. The last step in the classification process is to train the classifier.
We split the dataset into training and testing sets. For instance, in the experiments presented subsequently, we use 5-fold cross validation, so in each validation around 80% of the dataset is used for training and 20% for testing.

5. IMPLEMENTATION

The Implementation of the project involves

- ✓ Pre-processing
- ✓ Feature extraction
- ✓ Training classifier

5.1 PRE-PROCESSING

Data cleaning and preprocessing functions are required before extracting N-grams to reduce the text feature size. The dataset was cleaned by removing tweet ID, tweet time, hyperlinks, emoticons, punctuations and non-letter characters. Then preprocessing functions like stop word removal, stemming and tokenizing were done to remove trivial data and reduce the size of the actual data. Stop words are the words which occur commonly across all the tweets but actually they are insignificant like; a, an, the, will, was, were, is, are, to, of, that, these, what, when etc. These words must be removed because they are not discriminant when used as features in the classification task. Stemming is the process of removing suffixes and reduce words to their word stem. For example, words like (connects, connected, connecting, connection) all have the same meaning. Removing the suffixes (-ed, -s, -ion, -ing) and leaving the single word (connect) will reduce the number of unique words and make classification more efficient.

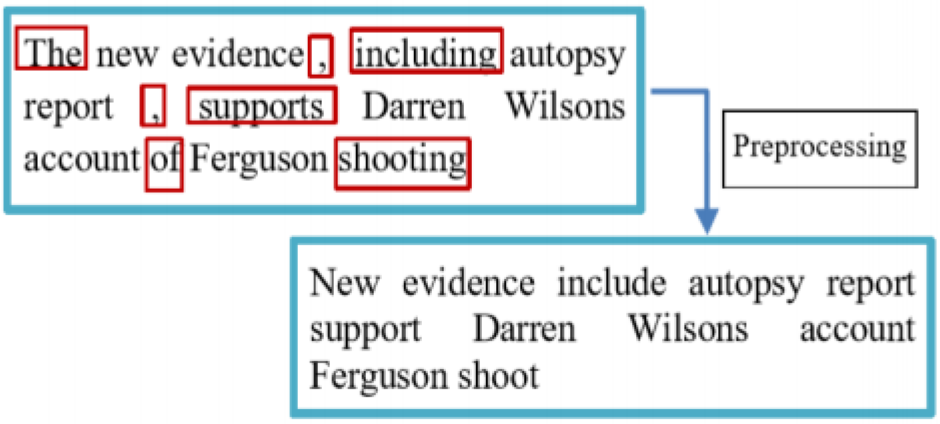

The new evidence , including autopsy report , supports Darren Wilsons account of Ferguson shooting
Preprocessing
New evidence include autopsy report support Darren Wilsons account Ferguson shoot

Fig 5.1.1 Preprocessing example

5.2 FEATURES EXTRACTION :

Features extraction is the process of obtaining the most relevant information from the original data forming the feature vectors. Most of the machine learning algorithms cannot accept the raw text data as input because they expect numerical feature vectors with a fixed size. Vectorization is the process of turning a collection of text documents into numerical feature vectors to represent them into a lower dimensionality space. This process consists of two stages: First tokenizing strings and giving an integer id for each possible token, then weighting the tokens or terms to represent the importance of each token. In this study, we used two different weighting methods, namely, Term Frequency (TF) and Term Frequency Inverted Document Frequency (TF-IDF).

Term frequency is simply assigning the weight to be equal to the number of occurrences of word w in tweet t. In this scheme, tweets are described by word occurrences where the word that occurs frequently is rewarded with completely ignoring the relative position of the words in the tweet.

TF-IDF is used in machine learning and text mining as a weighting factor for features. The weight increases as the word frequency in a document increases but that is offset by the number of times that word appears in the dataset. This mechanism helps to remove the importance from really common words that appear frequently in all documents and rewards words that are rare overall in a dataset. High TF-IDF weight is reached when a word has high TF in any given tweet and low DF of the word in the entire dataset. TF-IDF of a word in any given tweet is the product of the TF of this word and the inverse document frequency IDF of the word where,

$$\mathrm{IDFw} = \log 1+N \; 1+dfw + 1$$

where document frequency dfw is the number of tweets containing a word w in the entire dataset and N is the number of tweets.

N-gram model is commonly used in natural language processing applications. N-grams are sequences of words or characters as they appear in texts one after another where "N" corresponds to the number of elements in a sequence. The most used Ngram models in text analysis are word-based and character-based N-grams. In this work, we used word-based N-grams with n=1 (unigrams) which is known as the bag of words BOW, n=2 (bigrams) and n=3 (trigrams). For example, given this tweet ("Great Pyramids of Egypt"), the word-based unigrams (n=1) are (Great, Pyramids, Egypt) while bigrams are: (Great Pyramids, Pyramids of, of Egypt) and so on. The idea is to generate various sets of N-gram frequencies from the training data to represent the collected tweets. When using word-based N-gram analysis, determining the perfect value of N is an area of research. We used different values of N to generate N-gram features and examine the effect of the Ngram length on the accuracy of the different classification algorithms.

Our task is to predict whether a tweet is credible or not depending on the presence or absence of the most discriminative words related to it. For this task, we used the unigram model to compute the frequency for each word on the training set. Unigram model ignores the complete context of the word, so we extend our model by using bigrams, trigrams, and quad-grams. We observe that the unigram model achieved good results but using a combination of unigrams and bigrams leads to better performance.

5.3 MACHINE LEARNING CLASSIFIERS:

Natural Language Processing : The main reason for utilizing Natural Language Processing is to consider one or more specializations of system or an algorithm. The Natural Language Processing (NLP) rating of an algorithmic system enables the combination of speech understanding and speech generation. In addition, it could be utilized to detect actions with various languages. It suggested a new ideal system for extraction actions from languages of English, Italian and Dutch speeches through utilizing various pipelines of various languages such as Emotion Analyzer and Detection, Named Entity Recognition (NER), Parts of Speech (POS) Taggers, Chunking, and Semantic Role Labeling made NLP good Subject of the search.

Classifier : We compared the performance of five different supervised classifiers namely: Linear Support Vector Machines (LSVM), Logistic Regression (LR), Random Forests (RF), Naïve Bayes (NB) and K-Nearest Neighbor (KNN) in order to choose the best classifier. we evaluated the five classifiers with the two feature representations which we have extracted (TF/TF-IDF) in two separated experiments. We chose decision tree classifier for our project.

A decision tree is a set of decision nodes with a tree graph structure, like a flowchart. To make a decision, we start at the root. At each decision node, the input (information about the news article) is tested, and based on the outcome, we go along a branch to the next node. If the node is another decision node, we are again passed to a branch depending on the test outcome.

If the node is a leaf node, that is the classification that the decision tree has arrived at (real news or fake news). Pros of decision trees are, features can be dependent, classes do not need to be linearly separable, outliers are handled well, and the decision tree itself is easy to interpret.

One of the most widely used classifiers is Decision Tree Classifier. It is also a powerful classifier. Similar to SVM, Decision Tree can also perform both regression and classification. It is also a supervised learning algorithm. Decision Tree classifiers are more popular because tree analysis is easy to understand. It divides the given data set into small parts and a decision tree is incrementally constructed. The leaf nodes of a decision tree represent the classification. Decision trees are comfortable with numeric and categorical data.

DECISION TREE ALGORITHM

```
Decision Tree Pseudo-code GenerateDecisionTree(Sample s, features F)
1. If stop _conditions(S,F) = true then
a. leaf = create_Node()
b. Leaf.lable= classify(s)
c. Return leaf
2. root = create_Node()
3. root.testcondition = find_bestSplit(s,f)
4. v = { v l v a possible outcome of root.testconditions)
5. for each value v∈ V:
6. sv: = {s | root.testcondition(s) = v and s∈ S};
7. child = Tree_Growth(Sv ,F) ;
8. Grow child as a descent of roof and label the edge (root→child) as v

Return root
```

5.4 SOURCE CODE

```
#pandas is a software library written for the Python programming language for data manipulation and analysis.

from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
import matplotlib.pyplot as plt
import numpy as np
from tkinter import ttk
from tkinter import filedialog
import pandas as pd
from sklearn.model_selection import train_test_split
from string import punctuation
from nltk.corpus import stopwords
import nltk
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
import pickle



main = Tk()
main.title("FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM
ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE")
main.geometry("1300x1200")

global filename
global X, Y
global tfidf_X_train, tfidf_X_test, tfidf_y_train, tfidf_y_test
global tfidf_vectorizer
global accuracy,error
```

```
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

textdata = []
labels = []
global classifier

def cleanPost(doc):
    tokens = doc.split()
    table = str.maketrans('', '', punctuation)
    tokens = [w.translate(table) for w in tokens]
    tokens = [word for word in tokens if word.isalpha()]
    tokens = [w for w in tokens if not w in stop_words]
    tokens = [word for word in tokens if len(word) > 1]
    tokens = [lemmatizer.lemmatize(token) for token in tokens]
    tokens = ' '.join(tokens)
    return tokens

def uploadDataset():
    global filename
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir="TwitterNewsData")
    textdata.clear()
    labels.clear()
    dataset = pd.read_csv(filename)
    for i in range(len(dataset)):
        msg = dataset.get_value(i, 'text')
        label = dataset.get_value(i, 'target')
        msg = str(msg)
        msg = msg.strip().lower()
        labels.append(int(label))
        clean = cleanPost(msg)
        textdata.append(clean)
        text.insert(END,clean+" ==== "+str(label)+"\n")
```

```
def preprocess():
    text.delete('1.0', END)
    global Y
    global tfidf_vectorizer
    global tfidf_X_train, tfidf_X_test, tfidf_y_train, tfidf_y_test
    stopwords=stopwords = nltk.corpus.stopwords.words("english")
    tfidf_vectorizer = TfidfVectorizer(stop_words=stopwords, use_idf=True,
ngram_range=(1,2),smooth_idf=False, norm=None, decode_error='replace',
max_features=200)
    tfidf = tfidf_vectorizer.fit_transform(textdata).toarray()
    df = pd.DataFrame(tfidf, columns=tfidf_vectorizer.get_feature_names())
    text.insert(END,str(df))
    print(df.shape)
    df = df.values
    X = df[:, 0:200]
    Y = np.asarray(labels)
    le = LabelEncoder()
    Y = le.fit_transform(Y)
    indices = np.arange(X.shape[0])
    np.random.shuffle(indices)
    X = X[indices]
    Y = Y[indices]
    tfidf_X_train, tfidf_X_test, tfidf_y_train, tfidf_y_test = train_test_split(X, Y,
test_size=0.2)
    text.insert(END,"\n\nTotal News found in dataset : "+str(len(X))+"\n")
    text.insert(END,"Total records used to train machine learning algorithms :
"+str(len(tfidf_X_train))+"\n")
    text.insert(END,"Total records used to test machine learning algorithms  :
"+str(len(tfidf_X_test))+"\n")
```

```
  def runDecisionTree():
     text.delete('1.0', END)
     global classifier
     global accuracy,error
     cls = DecisionTreeClassifier()
     cls.fit(tfidf_X_train, tfidf_y_train)
     predict = cls.predict(tfidf_X_test)
     acc = accuracy_score(tfidf_y_test,predict)*100
     accuracy = acc
     error = 100 - accuracy
     text.insert(END,"Decision Tree Accuracy : "+str(acc)+"\n")
     classifier = cls
     with open('model.txt', 'wb') as file:
        pickle.dump(classifier, file)
     file.close()



  def graph():
     bars = ('Decision Tree Accuracy','Decision Tree Error Rate')
     y_pos = np.arange(len(bars))
     plt.bar(y_pos, [accuracy,error])
     plt.xticks(y_pos, bars)
     plt.show()


def predict():
   testfile = filedialog.askopenfilename(initialdir="TwitterNewsData")
   testData = pd.read_csv(testfile)
   text.delete('1.0', END)
   testData = testData.values
```

```
    testData = testData[:,0]
    print(testData)
    for i in range(len(testData)):
        msg = testData[i]
        msg1 = testData[i]
        print(msg)
        review = msg.lower()
        review = review.strip().lower()
        review = cleanPost(review)
        testReview = tfidf_vectorizer.transform([review]).toarray()
        predict = classifier.predict(testReview)
        print(predict)
        if predict == 0:
            text.insert(END,msg1+" === Given news predicted as GENUINE\n\n")
        else:
            text.insert(END,msg1+" == Given news predicted as FAKE\n\n")

font = ('times', 15, 'bold')
title = Label(main, text='FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME,
N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE')
#title.config(bg='powder blue', fg='olive drab')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)

font1 = ('times', 13, 'bold')
ff = ('times', 12, 'bold')

uploadButton = Button(main, text="Upload Fake News Dataset", command=uploadDataset)
uploadButton.place(x=20,y=100)
uploadButton.config(font=ff)
processButton = Button(main, text="Preprocess Dataset & Apply NGram",
command=preprocess)
processButton.place(x=20,y=150)
processButton.config(font=ff)

dtButton = Button(main, text="Run Decision Tree Algorithm", command=runDecisionTree)
dtButton.place(x=20,y=200)
dtButton.config(font=ff)

graphButton = Button(main, text="Decision Tree Accuracy & Error Rate Graph",
command=graph)
graphButton.place(x=20,y=250)
graphButton.config(font=ff)

predictButton = Button(main, text="Test News Detection", command=predict)
predictButton.place(x=20,y=300)
predictButton.config(font=ff)

font1 = ('times', 12, 'bold')
text=Text(main,height=30,width=85)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=450,y=100)
text.config(font=font1)

main.config()
main.mainloop()
```

6. TESTING

OUTPUT SCREEN

- To run project double click on ‘run.bat’ file to get below screen

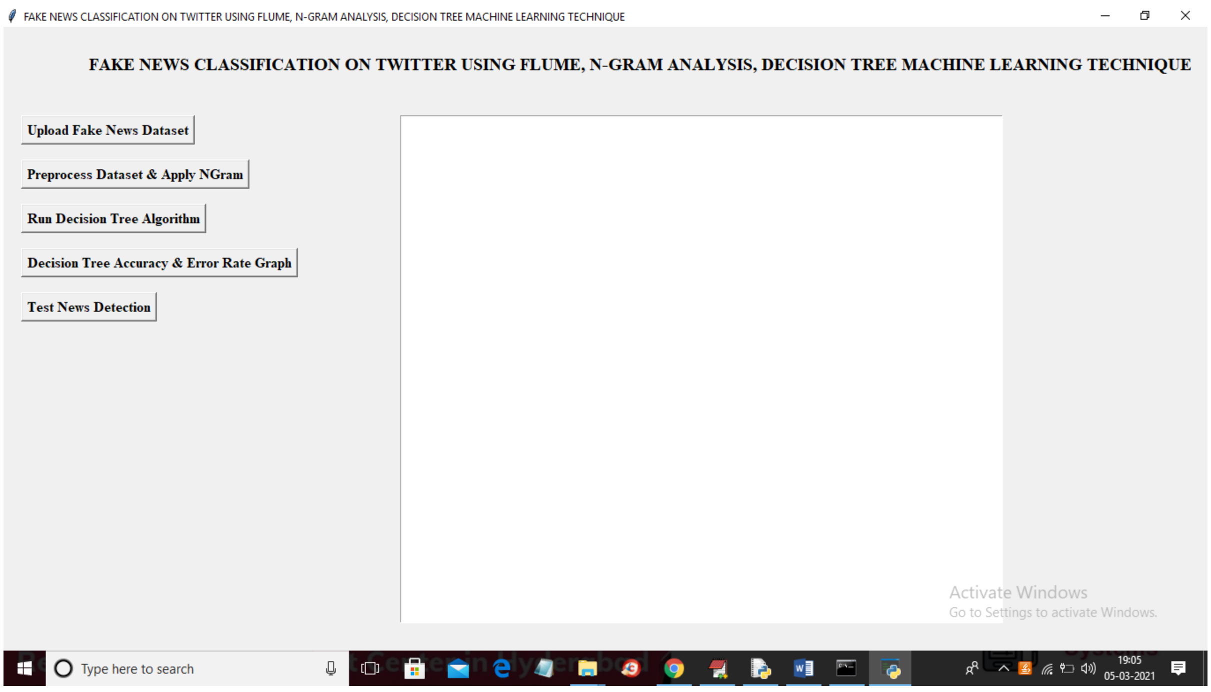

FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE
FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE
Upload Fake News Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
Decision Tree Accuracy & Error Rate Graph
Test News Detection
Activate Windows
Go to Settings to activate Windows.
Type here to search
19:05
05-03-2021

Fig 6.1 Running the project

- In above screen click on ‘Upload Fake News Dataset’ button to upload dataset

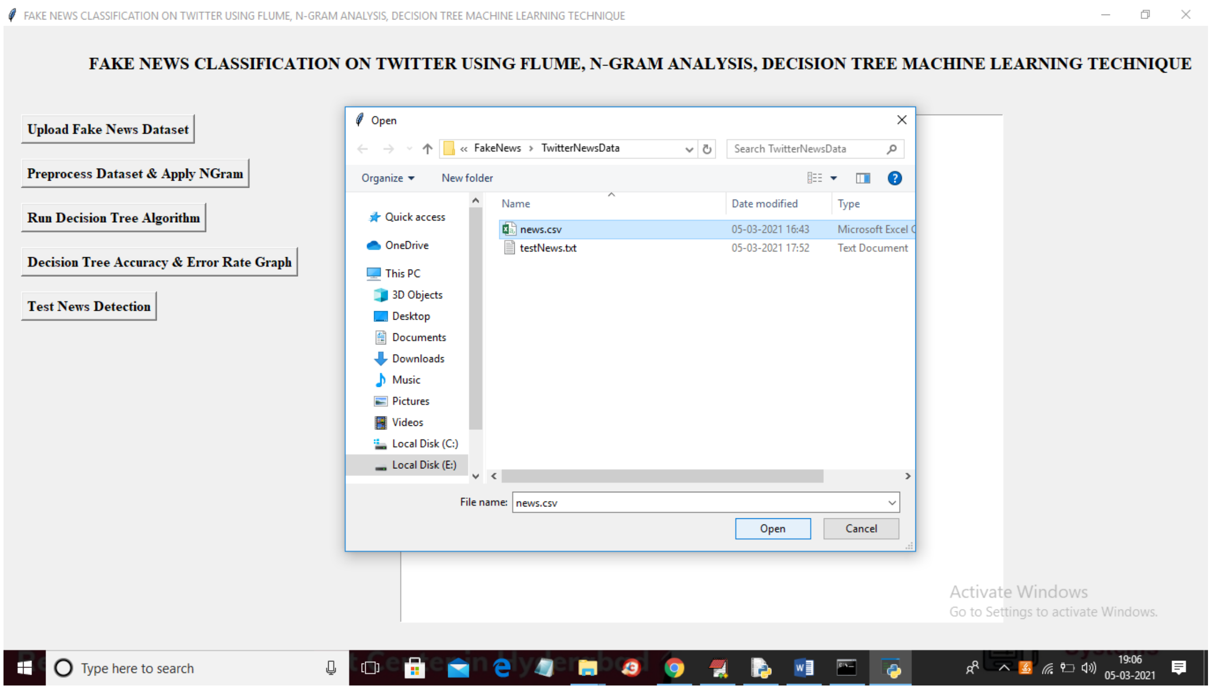
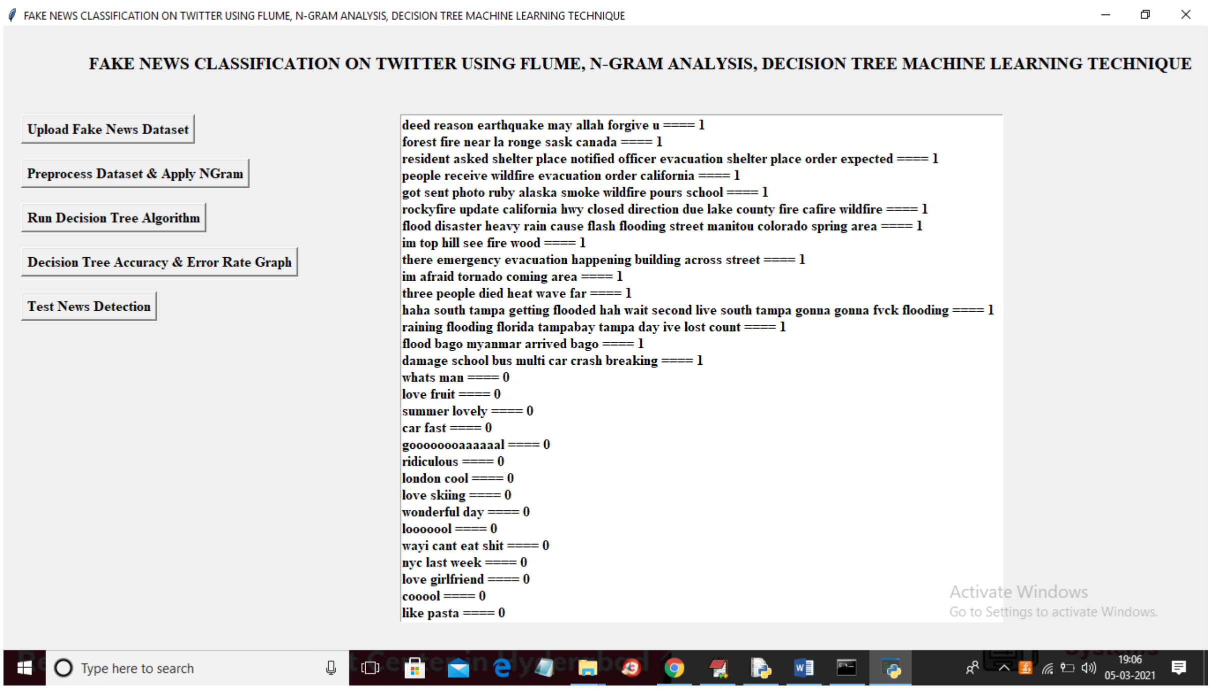

FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE
Upload Fake News Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
Decision Tree Accuracy & Error Rate Graph
Test News Detection
Open
FakeNews > TwitterNewsData
news.csv
testNews.txt
File name: news.csv
Open
Cancel

Fig 6.2 Uploading dataset

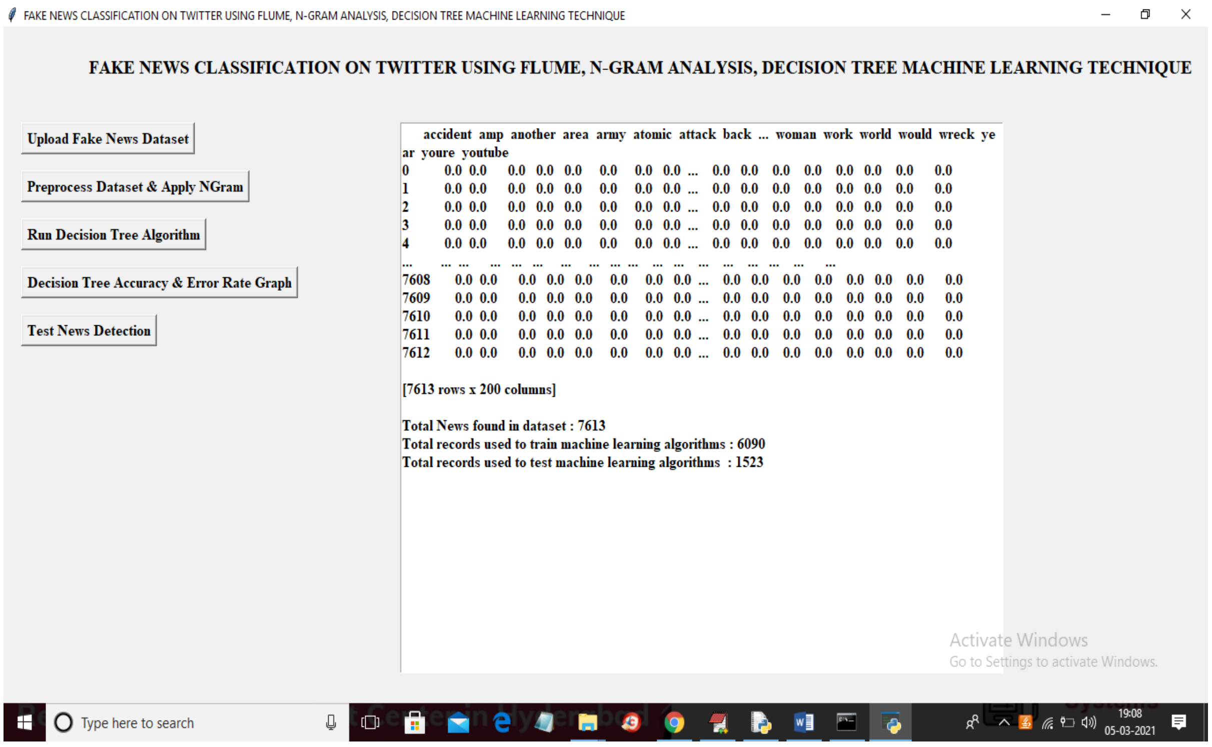
- In above screen selecting and uploading 'news.csv' file and then click on 'Open' button to load dataset and to get below screen


FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE
Upload Fake News Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
Decision Tree Accuracy & Error Rate Graph
Test News Detection

```
deed reason earthquake may allah forgive u ==== 1
forest fire near la ronge sask canada ==== 1
resident asked shelter place notified officer evacuation shelter place order expected ==== 1
people receive wildfire evacuation order california ==== 1
got sent photo ruby alaska smoke wildfire pours school ==== 1
rockyfire update california hwy closed direction due lake county fire cafire wildfire ==== 1
flood disaster heavy rain cause flash flooding street manitou colorado spring area ==== 1
im top hill see fire wood ==== 1
there emergency evacuation happening building across street ==== 1
im afraid tornado coming area ==== 1
three people died heat wave far ==== 1
haha south tampa getting flooded hah wait second live south tampa gonna gonna fvck flooding ==== 1
raining flooding florida tampabay tampa day ive lost count ==== 1
flood bago myanmar arrived bago ==== 1
damage school bus multi car crash breaking ==== 1
whats man ==== 0
love fruit ==== 0
summer lovely ==== 0
car fast ==== 0
goooooooaaaaaal ==== 0
ridiculous ==== 0
london cool ==== 0
love skiing ==== 0
wonderful day ==== 0
looooool ==== 0
wayi cant eat shit ==== 0
nyc last week ==== 0
love girlfriend ==== 0
cooool ==== 0
like pasta ==== 0
```

Fig 6.3 Labelled dataset

- In above screen dataset loaded and then in text area we can see all news text with the class label as 0 or 1 and now click on 'Preprocess Dataset & Apply NGram' button to convert above string data to numeric vector and to get below screen.


FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE
FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE
Upload Fake News Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
Decision Tree Accuracy & Error Rate Graph
Test News Detection

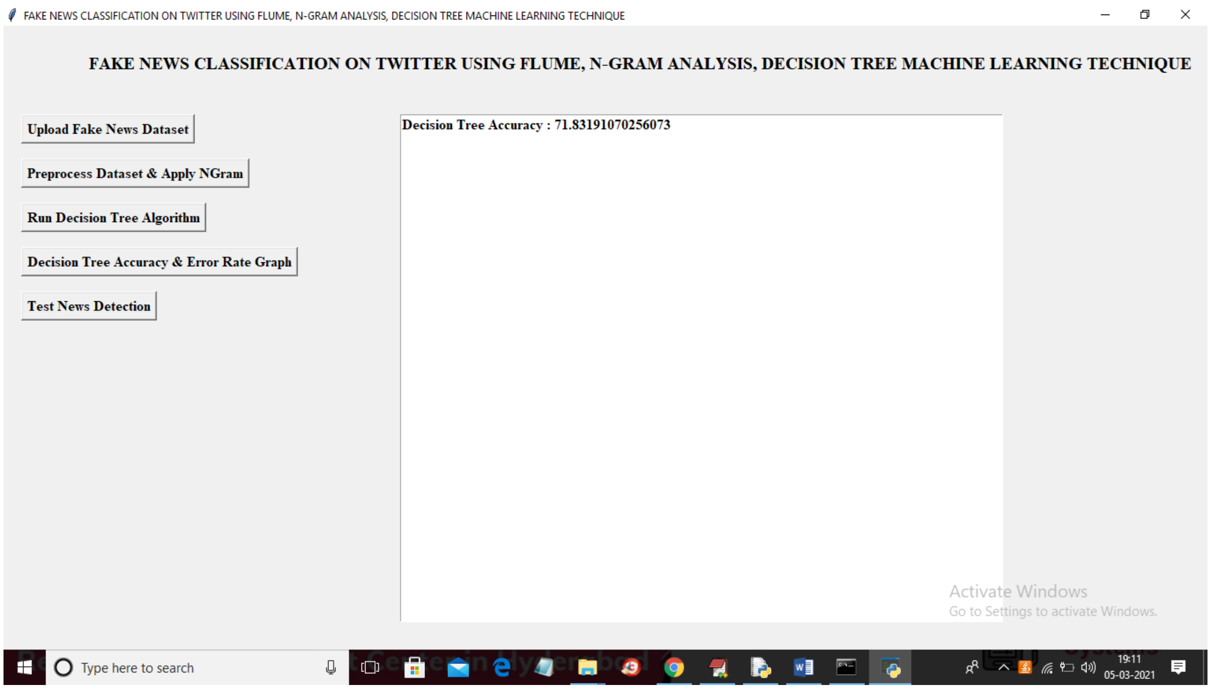
```
    accident  amp  another  area  army  atomic  attack  back  ...  woman  work  world  would  wreck  ye
ar  youre  youtube
0        0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
1        0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
2        0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
3        0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
4        0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
...      ...  ...     ...   ...   ...     ...     ...   ...  ...    ...   ...    ...    ...    ...   ...    ...      ...
7608     0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
7609     0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
7610     0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
7611     0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0
7612     0.0  0.0     0.0   0.0   0.0     0.0     0.0   0.0  ...    0.0   0.0    0.0    0.0    0.0   0.0    0.0      0.0

[7613 rows x 200 columns]

Total News found in dataset : 7613
Total records used to train machine learning algorithms : 6090
Total records used to test machine learning algorithms  : 1523
```

Fig 6.4 Records

- In above screen all news words put in column header and if that word appear in any row then that rows column will be change with word count and if not appear then 0 will be put in column. In above screen showing some records from total 7612 news records and in bottom lines we can see dataset contains total 7613 records and then application using 80% (6090 news records) for training and then using 20% (1523 news records) for testing and now dataset is ready with numeric record and now click on 'Run Decision Tree Algorithm' button to train above dataset with decision tree and then build decision tree model and then calculate accuracy and error rate.

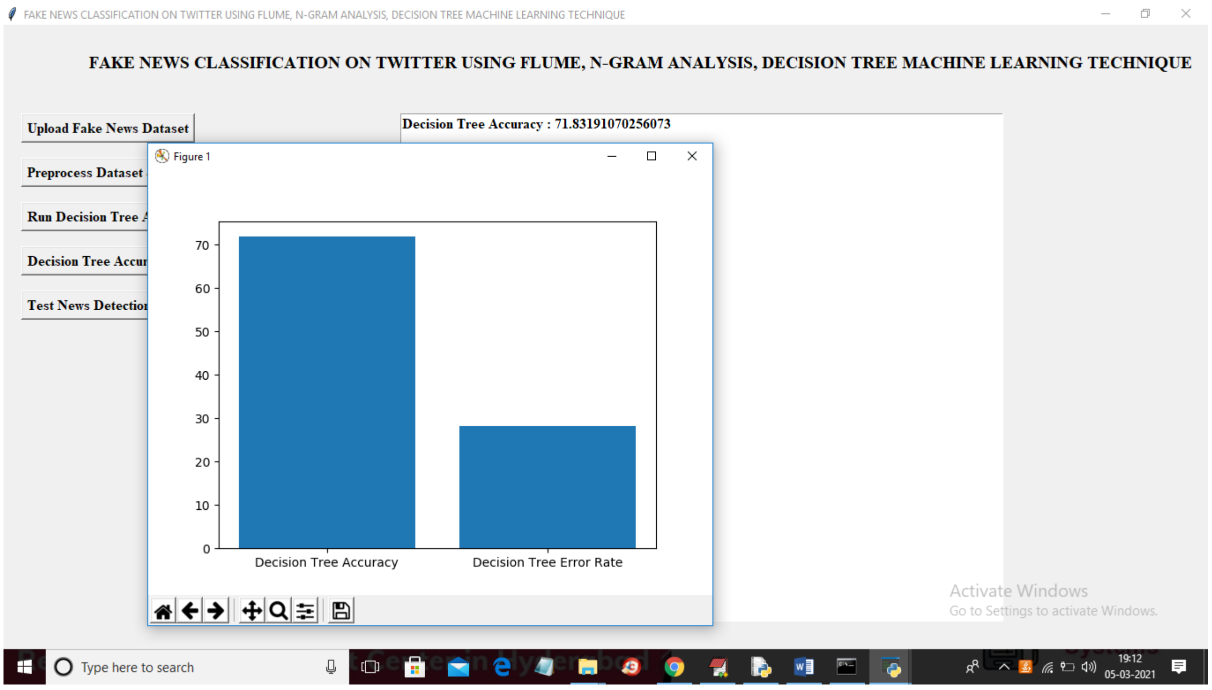

FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE

Decision Tree Accuracy : 71.83191070256073

Upload Fake News Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
Decision Tree Accuracy & Error Rate Graph
Test News Detection

Fig 6.5 Decision Tree Accuracy

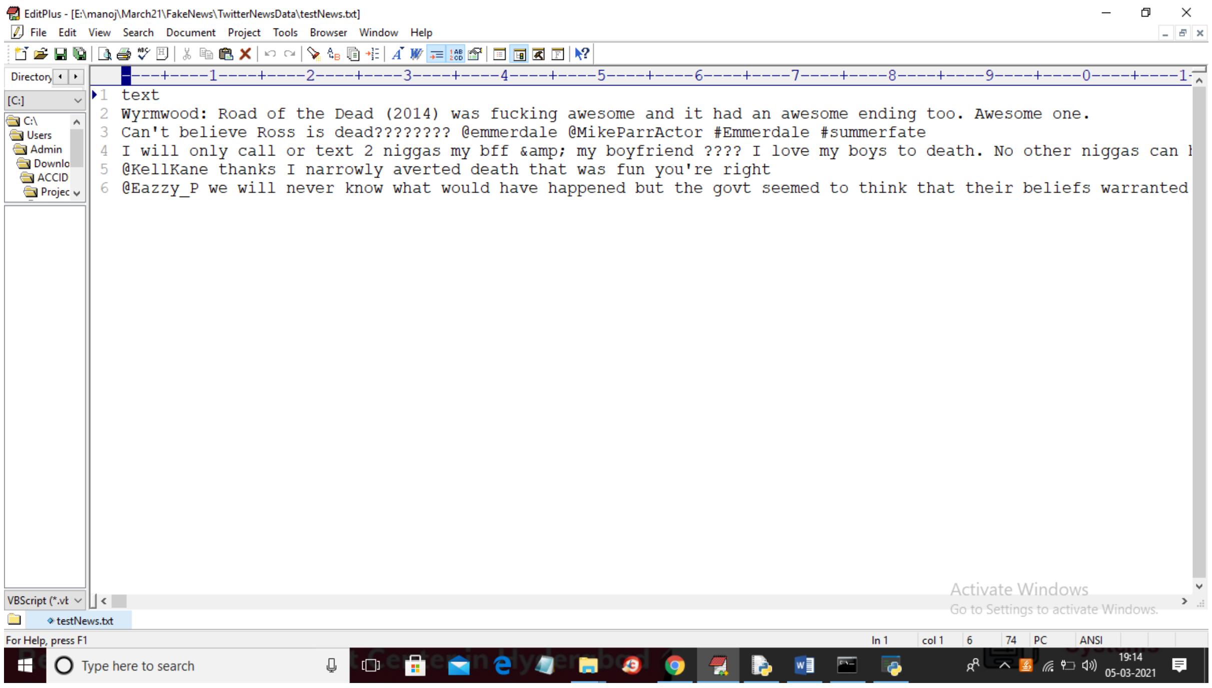
In above screen decision tree model is generated and we got its prediction accuracy as 71% and now click on 'Decision Tree Accuracy & Error Rate Graph' button to get below graph


FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE

Decision Tree Accuracy : 71.83191070256073

Fig 6.6 Graph

- In above graph x-axis represents decision tree accuracy and its error rate and y-axis represents accuracy and error rate value and from above graph we can conclude that decision tree correct prediction accuracy is more than error rate. Now click on ‘Test News Detection’ button to upload some test news sentences and then application predict whether that news is genuine or fake. In below test news dataset we can see only TEXT data no class label and decision tree will predict class label for that test news


EditPlus - [E:\manoj\March21\FakeNews\TwitterNewsData\testNews.txt]
File Edit View Search Document Project Tools Browser Window Help
testNews.txt
For Help, press F1

```
text
Wyrmwood: Road of the Dead (2014) was fucking awesome and it had an awesome ending too. Awesome one.
Can't believe Ross is dead???????? @emmerdale @MikeParrActor #Emmerdale #summerfate
I will only call or text 2 niggas my bff &amp; my boyfriend ???? I love my boys to death. No other niggas can 
@KellKane thanks I narrowly averted death that was fun you're right
@Eazzy_P we will never know what would have happened but the govt seemed to think that their beliefs warranted
```

Fig 6.7 Test News

- In above screen in test new we have only one column which contains only news ‘TEXT’ and after applying above test news we will get prediction result

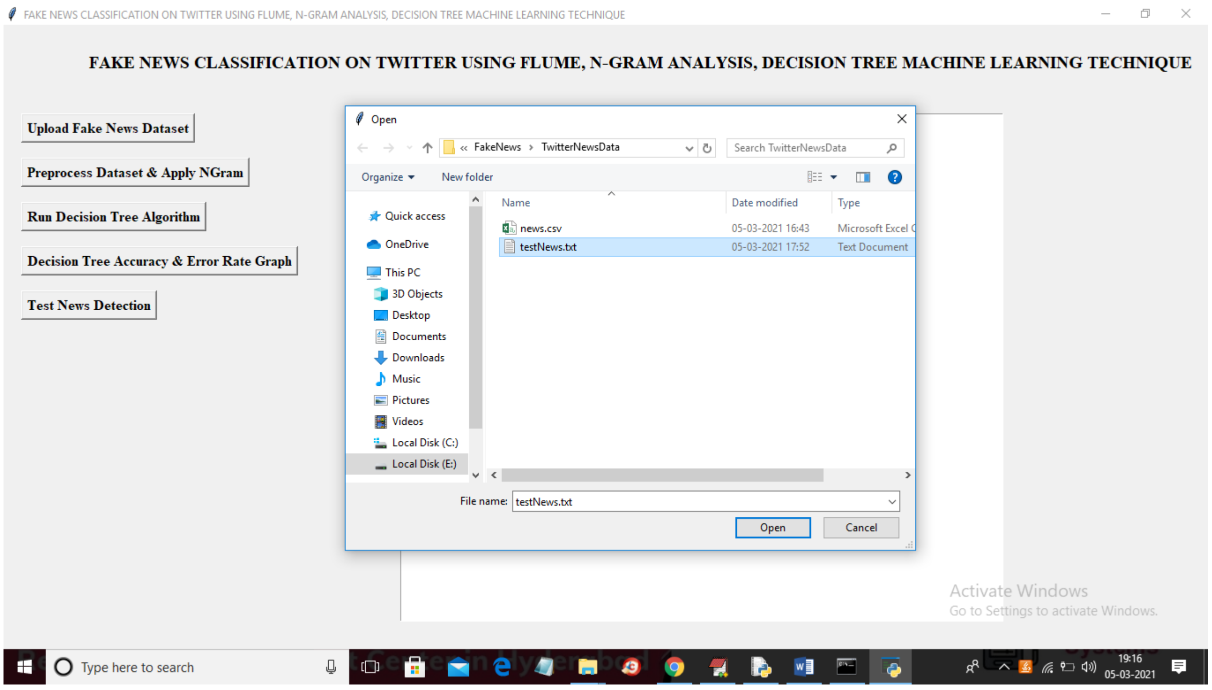
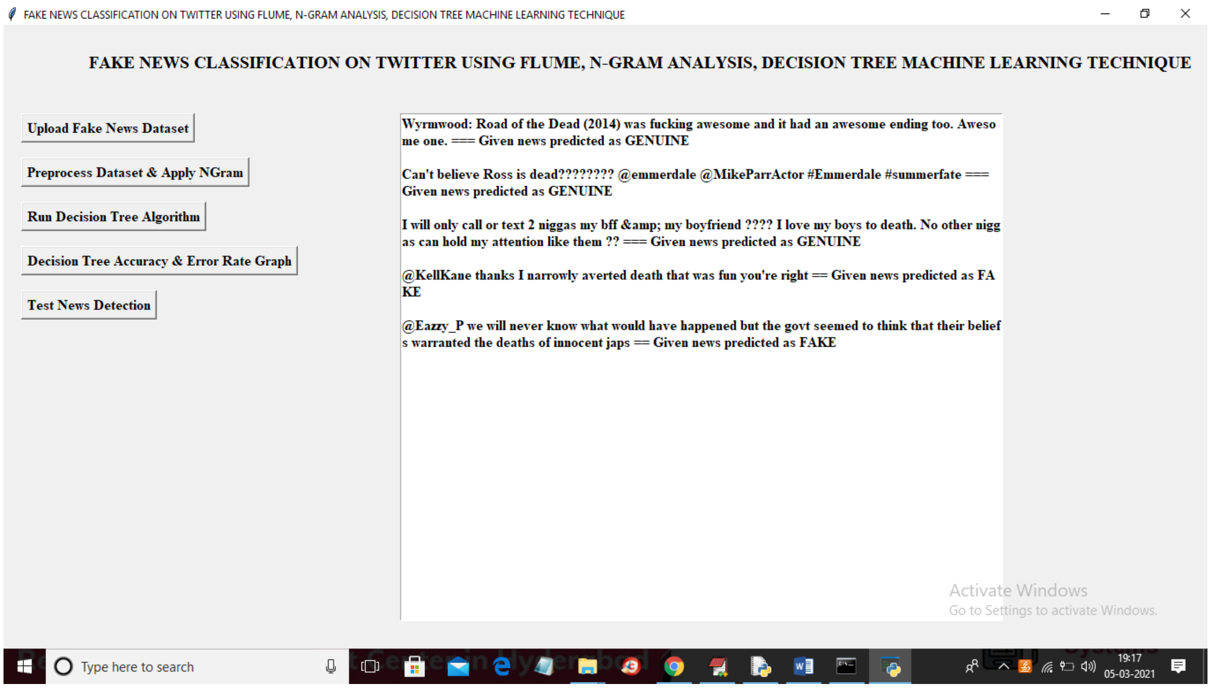


Fig 6.8 Uploading ‘testNews.txt’ file

- In above screen selecting and uploading ‘testNews.txt’ file and then click on ‘Open’ button to load data and to get below prediction result


FAKE NEWS CLASSIFICATION ON TWITTER USING FLUME, N-GRAM ANALYSIS, DECISION TREE MACHINE LEARNING TECHNIQUE
Upload Fake News Dataset
Preprocess Dataset & Apply NGram
Run Decision Tree Algorithm
Decision Tree Accuracy & Error Rate Graph
Test News Detection
Type here to search

```
Wyrmwood: Road of the Dead (2014) was fucking awesome and it had an awesome ending too. Aweso
me one. === Given news predicted as GENUINE

Can't believe Ross is dead???????? @emmerdale @MikeParrActor #Emmerdale #summerfate ===
Given news predicted as GENUINE

I will only call or text 2 niggas my bff &amp; my boyfriend ???? I love my boys to death. No other nigg
as can hold my attention like them ?? === Given news predicted as GENUINE

@KellKane thanks I narrowly averted death that was fun you're right == Given news predicted as FA
KE

@Eazzy_P we will never know what would have happened but the govt seemed to think that their belief
s warranted the deaths of innocent japs == Given news predicted as FAKE
```

Fig 6.9 Result

- In above screen before dashed symbols we have news text and after dashed symbol application predict news as ‘FAKE or GENUINE’. After building model when we gave any news text then decision tree will check whether more words belongs to genuine or fake category and whatever category get more matching percentage then application will predict that class label.

1. CONCLUSION

.

In this work, we focused on the problem of detecting the credibility of Twitter messages using text-based features. We have presented a credibility detection model based on N-gram analysis and investigated two different feature extraction techniques. The obtained results indicated that word N-gram features are more relevant compared with content and source-based features, also compared with character N-gram features. Linear classifiers as LSVM and logistic regression are more suitable for this problem. Best results were achieved using a combination of unigrams and bigrams, 30000 TFIDF extracted features and LSVM as a classifier. The proposed model achieved 84.9% accuracy, 86.6% precision, 91.9% recall, and 89% F-measure over the PHEME dataset. For the CAT dataset, the model achieved 73.2% Accuracy, 76.4% Precision, 80.7% Recall, and 78.5% F-Measure. The evaluation shows higher performance of the proposed model in comparison with three different models existing in the literature using the same dataset.

2. FUTURE SCOPE

As a future work, we plan to train the N-gram model on a larger dataset to increase the robustness of the model. We think that applying dimensionality reduction techniques to keep only the most relevant features from 30000 features set will improve the model performance.. Regarding the online mobile application, we think it will be better if the output is presented to the user as a continuous score instead of the binary classification.

3. REFERENCES

- Z. Jin, J. Cao, Y. Jiang, and Y. Zhang. "News credibility evaluation on microblog with a hierarchical propagation model." In 2014 IEEE International Conference on Data Mining, pp. 230-239. IEEE, 201

- H. Al-Khalifa, and R. Al-Eidan. "An experimental system for measuring the credibility of news content in Twitter.". International Journal of Web Information Systems Vol. 7, No. 2, pp.130-151, 2011.

- SCIKIT-LEARN Machine Learning in Python http://scikit-learn.org/stable/

- “Twitter by the numbers: Stats, Demographic & Fun Facts.” [Online]. Available: https://www.omnicoreagency.com/twitterstatistics/ [Accessed: 17-10-2019]

- “Twitter by the numbers: Stats, Demographic & Fun Facts.” [Online]. Available: https://www.omnicoreagency.com/twitterstatistics/ [Accessed: 17-10-2019] twitter

- H. Allcott, and M. Gentzkow. "Social media and fake news in the 2016 election." Journal of economic perspectives Vol. 31, No. 2, pp. 211- 36, 2017.

- SCIKIT-LEARN Machine Learning in Python http://scikit-learn.org/stable/

- S. Ravikumar, R. Balakrishnan, and S. Kambhampati. "Ranking tweets considering trust and relevance." In Proceedings of the Ninth International Workshop on Information Integration on the Web, p. 4. ACM, 2012.