ATMIYA UNIVERSITY

RAJKOT

A

Report On

Fake News Prediction

Under subject of

Major Project

B. TECH, Semester– VII

(Computer Engineering)

Submitted by:

1. Arjunsinh Zala 190002127

Prof. Ambrish Patel

(Faculty Guide)

Prof. Tosal M.Bhalodia

(Head of the Department)

Academic Year

(2022-23)

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this project entitled "**Fake News Prediction"** submitted towards completion of project in **7th Semester** of B.Tech. (Computer Engineering) is an authentic record of our original work carried out under the guidance of "**Prof. Ambrish Patel".**

I have not submitted the matter embodied in this project for the award of any other degree.

Semester: 7th
Place: Rajkot

Signature:
Arjunsinh Ajitsinh Zala
(190002127)

ATMIYA UNIVERSITY

RAJKOT

CERTIFICATE

Date:

This is to certify that the "**Fake News Prediction**" has been carried out by **Arjunsinh Zala** under my guidance in fulfillment of the subject Project in COMPUTER ENGINEERING (7thSemester) of Atmiya University, Rajkot during Academic year 2022-23.

Prof. Ambrish Patel — **(Project Guide)**

Prof.Tosal M.Bhalodia — **(Head of the Department)**

ACKNOWLEDGEMENT

I have taken many efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere gratitude to all of them.

I am highly indebted to Prof. Ambrish Patel for her guidance and constant supervision as well as for providing necessary information regarding the Major Project Titled **"Fake News Prediction".** I would like to express My gratitude towards staff members of Computer Engineering Department, Atmiya University for their kind co- operation and encouragement which helped me in completion of this project.

I even thank and appreciate to My colleagues in developing the project and people who have willingly helped me out with their abilities.

Arjunsinh Zala (190002127)

ABSTRACT

In today’s worlds where people are more reliable on the news which are available online as it's convenient for them. As the use of the internet is increasing so thus the spread of fake news also. As the spread of such fake news can be intentional or unintentional but this affects society. Thus, an increasing number of fake news has to be controlled by using the computational tool which predicts such misleading information as if it is fake or real.

INDEX

Figure No.	List Of Figures	Page No.

CHAPTER 1:
INTRODUCTION

1.1 Introduction

These days' fake news is creating different issues from sarcastic articles to a fabricated news and plan government propaganda in some outlets. Fake news and lack of trust in the media are growing problems with huge ramifications in our society. Obviously, a purposely misleading story is "fake news" but lately blathering social media's discourse is changing its definition. Some of them now use the term to dismiss the facts counter to their preferred viewpoints. Fortunately, there are a number of computational techniques that can be used to mark certain articles as fake on the basis of their textual content. As human beings, when we read a sentence or a paragraph, we can interpret the words with the whole document and understand the context. In this project, we teach a system how to read and understand the differences between real news and the fake news using concepts like natural language processing, NLP and machine learning and prediction classifiers like the Logistic regression and multinomial Naïve bayes which will predict the truthfulness or fake-news of an article.

1.2 Purpose

Due to the exponential growth of information online, it is becoming impossible to decipher the true from the false. Thus, this leads to the problem of fake news. The fake news for various commercial and political purposes has been emerging in large numbers and widely spread in the online world. The existing systems are not efficient in giving a precise statistical rating for any given news. Also, the restrictions on input and category of news make it less varied.so we will develop a method for automating fake news detection for various events. We'll use Logistic Regression language processing techniques to classify fake news.

1.3 Scope

Future Scope of this project is vast as use of internet is increasing day by day and sharing of article, news via social media, messaging platform is increasing at vast level. To control such incidents this model could be used which shows the user about truthfulness of the message or post they intend to share to others. This will help society to avoid fake news.

1.4 Application Of Fake News Detection

Fake news detection can be used to prevent various types of fake news like:

1. Clickbait:
   Often eye-catching content to capture readers at the expense of being factual.

2. Satire/parody:
   This type of content is considered to be fun and humorous thus considered to be entertaining, yet some readers may interpret the content as fact.

3. Propaganda:
   This is content meant to mislead and influence the reader.

4. Biased/partisan/hyper-partisan:
   Oftentimes this is biased political content claiming to be impartial.

5. Unreliable news:
   Journalists may publish news whose sources are unverified, or without carrying out any form of fact checking themselves.

1.5 Technical feasibility

This involves questions such as whether the technology needed for the system exists, how difficult it will be to build, and whether the firm has enough experience using that technology. The assessment is based on outline design of system requirements in terms of input, processes, output, fields, programs and procedures. This can be qualified in terms of volume of data, trends, frequency

of updating in order to give an introduction to the technical system.

1.6 Language And Libraries

- **1.6.1 Language:**

  - Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.
  - It is used for:
    1) web development (server-side),
    2) software development,
    3) mathematics
    4) system scripting.

- **1.6.2 Why Python?:**

  - Python works on different platforms (Windows, Mac, Linux, RaspberryPi, etc.)
  - Python has a simple syntax similar to the English language.
  - Python has syntax that allows developers to write programs with few lines than some other programming languages.
  - Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
  - Python can be treated in a procedural way, an object-oriented way or a functional way.

- **1.6.3 Libraries Used:**

  1) NumPy

     - NumPy is a general-purpose array-processing package.
     - It provides a high-performance multidimensional array object, and tools for working with these arrays.
     - It is the fundamental package for scientific computing with Python.
     - Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data**.**

2) RegEx
   - A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern.

   - RegEx can be used to check if a string contains the specified search pattern.
   - Here its used for stemming.

3) Pandas
   - Pandas is an open-source library that is made mainly for working with relational or labelled data both easily and intuitively.
   - It provides various data structures and operations for manipulating numerical data and time series.
   - This library is built on top of the NumPy library.
   - Pandas is fast and it has high performance & productivity for users.

4) Natural Language Toolkit (NLTK)
   - NLTK is a leading platform for building Python programs to work with human language data.
   - Natural Language Processing with Python provides a practical introduction to programming for language processing.
   - It can be used for analysing and categorizing text, analysing linguistic structure

5) Scikit Learn
   - scikit-learn is a Python module for machine learning built on top of SciPy and is distributed under the 3-Clause BSD license.
   - Used for Classification, Regression, Clustering, Dimensionality Reduction, Model Selection, Pre-processing.

CHAPTER 2:
PROJECT MANAGEMENT

2.1 Project Planning

There is only a single member for development of this project, at first, I gathered information regarding My project like advantages, needs & other requirements, then I prepared a power point presentation of my project, then I prepared a project report and we started development of my project.

2.2 Project Scheduling

I decided to follow spiral model for developing this system for given reasons:

- Spiral model combines the advantages of top-down and bottom-up concepts.
- this system needs continuous development.
- I will describe the characteristics with high priority first and then develop a prototype based on this.
- This prototype will be tested and desired changes will be made in the new system.
- This continual and steady approach will minimize the risks or failure associated with the change in the system.
- The system will be developed in small segments that will make it easier to do cost calculations.
- The client will be involved in the development of each segment and retains

control over the direction and implementation of the system.

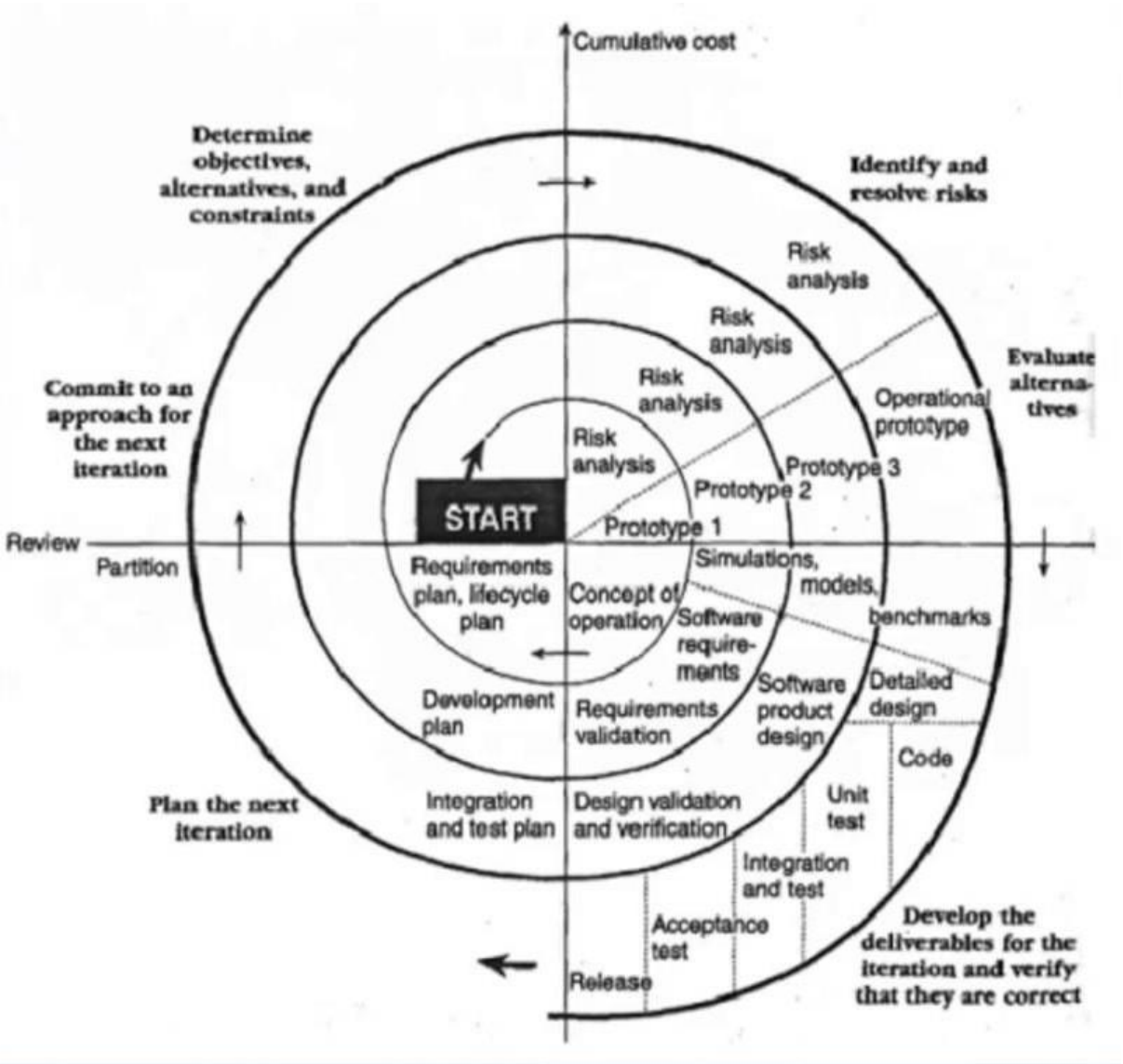


Fig 2.2.1 Spiral Model

CHAPTER 3:
SYSTEM REQUIREMENTS

3.1 Minimum Hardware Requirements

- 800 MHz Intel Pentium III or equivalent
- 256 MB of RAM
- 750 MB of free disk space
- Display (800 x 600-pixel resolution or higher)

3.2 Minimum Software Requirements

- Operating system compatible with python
- Python IDE (Google collab/Jupyter Notebook)

CHAPTER 4:

TESTING AND IMPLEMENTATION

The term implementation has different meanings ranging from the conversation of a basic application to a complete replacement of a computer system. The procedures however, are virtually the same. Implementation includes all those activities that take place to convert from old system to new. The new system may be totally new replacing an existing manual or automated system or it may be major modification to an existing system. The method of implementation and time scale to be adopted is found out initially. Proper implementation is essential to provide a reliable system to meet organization requirement.

4.1 Unit Testing

- **4.1.1 Introduction:**

  In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended.

- **4.1.2 Benefits**

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

1) Find problems early:
Unit testing finds problems early in the development cycle. In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is considered to be a bug either in the changed code or the tests themselves. The unit tests then allow the location of the fault or failure to be easily traced. Since the unit tests alert the development team of the problem before handing the code off to testers or clients, it is still early in the development process.

2) Facilitates Change:
Unit testing allows the programmer to refactor code or upgrade system libraries at a later date, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes which may break a design contract.

3) Simplifies Integration:
Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier.

4) Documentation:
Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface (API).Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviors that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

4.2 Integration Testing

Integration testing (a.k.a. integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

- **4.2.1 Purpose**

  The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using black-box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages. Software integration testing is performed according to the software development life cycle (SDLC) after module and functional tests. The cross-dependencies for software integration testing are: schedule for integration testing, strategy and selection of the tools used for integration, define the cyclomatic complexity of the software and software architecture, reusability of modules and life-cycle and versioning management. Some different types of integration testing are big-bang, top-down, and bottom-up, mixed (sandwich) and risky-hardest. Other Integration Patterns are: collaboration integration, backbone integration, layer integration, client-server integration, distributed services integration and high-frequency

integration.

- **4.2.1.1 Big Bang**
  In the big-bang approach, most of the developed modules are coupled together to form a complete software system or major part of the system and then used for integration testing. This method is very effective for saving time in the integration testing process. However, if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing. A type of big-bang integration testing is called "usage model testing" which can be used in both software and hardware integration testing. The basis behind this type of integration testing is to run user-like workloads in integrated user-like environments. In doing the testing in this manner, the environment is proofed, while the individual components are proofed indirectly through their use. Usage Model testing takes an optimistic approach to testing, because it expects to have few problems with the individual components. The strategy relies heavily on the component developers to do the isolated unit testing for their product. The goal of the strategy is to avoid redoing the testing done by the developers, and instead flesh-out problems caused by the interaction of the components in the environment. For integration testing, Usage Model testing can be more efficient and provides better test coverage than traditional focused functional integration testing. To be more efficient and accurate, care must be used in defining the user-like workloads for creating realistic scenarios in exercising the environment. This gives confidence that the integrated environment will work as expected for the target customers.

- **4.2.1.2 Top Down and Bottom Up**
  Bottom-up testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher-level components. The process is repeated until the component at the top of the hierarchy is tested. All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower-level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.

Top-down testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module is tested step by step until the end of the related module. Sandwich testing is an approach to combine top down testing with bottom up testing.

4.3 Software Verification and Validation

- **4.3.1 Introduction**

- In software project management, software testing, and software engineering, verification and validation (V&V) is the process of checking that a software system meets specifications and that it fulfills its intended purpose. It may also be referred to as software quality control. It is normally the responsibility of software testers as part of the software development lifecycle. Validation checks that the product design satisfies or fits the intended use (high-level checking), i.e., the software meets the user requirements. This is done through dynamic testing and other forms of review. Verification and validation are not the same thing, although they are often confused. Boehm succinctly expressed the difference between

- Validation: Are we building the right product?

  Verification: Are we building the product, right?

  According to the Capability Maturity Model (CMMI-SW v1.1)

  Software Verification: The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.

  Software Validation: The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.

  In other words, software verification is ensuring that the product has been built according to the requirements and design specifications, while

software validation ensures that the product meets the user's needs, and that the specifications were correct in the first place. Software verification ensures that "you built it right". Software validation ensures that "you built the right thing". Software validation confirms that the product, as provided, will fulfill its intended use.

From Testing Perspective

Fault – wrong or missing function in the code.
Failure – the manifestation of a fault during execution.
Malfunction – according to its specification the system does not meet its specified functionality
Both verification and validation are related to the concepts of quality and of software quality assurance. By themselves, verification and validation do not guarantee software quality; planning, traceability, configuration management and other aspects of software engineering are required.Within the modeling and simulation (M&S) community, the definitions of verification, validation and accreditation are similar:

M&S Verification is the process of determining that a • computer model, simulation, or federation of models and simulations implementations and their associated data accurately represent the developer's conceptual description and specifications.
M&S Validation is the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data are accurate representations of the real world from the perspective of the intended use(s).

CHAPTER 5:
SYSTEM DESIGN

5.1 Diagrams

The data flow diagram (DFD) is a graphical tool used for expressing system requirements in a graphical form. The DFD also known as the bubble chart as the purpose of clarification system requirements and identification major transformation that will become program in system design. Thus, DFD can be stated as the starting point of the design phase that functionality decomposes the requirements specification down to the lowest level of details. The DFD consists of series of bubble joined by lines. The bubble represents data transformation and the lines represents the data flows in the system. A DFD describes what data flow is does not to construct a Data Flow Diagram, we use

- Arrow: An arrow identifies the data flow in motion. It is a pipeline through which information is flow like the rectangle in the flowchart.
- Circle: A circle stands for process that converts data into information
- Open End Box: An open-ended box represents a data store, data at rest or a temporary repository of data
- Squares: A square defines a source or destination of system

5.1.1 Flowchart

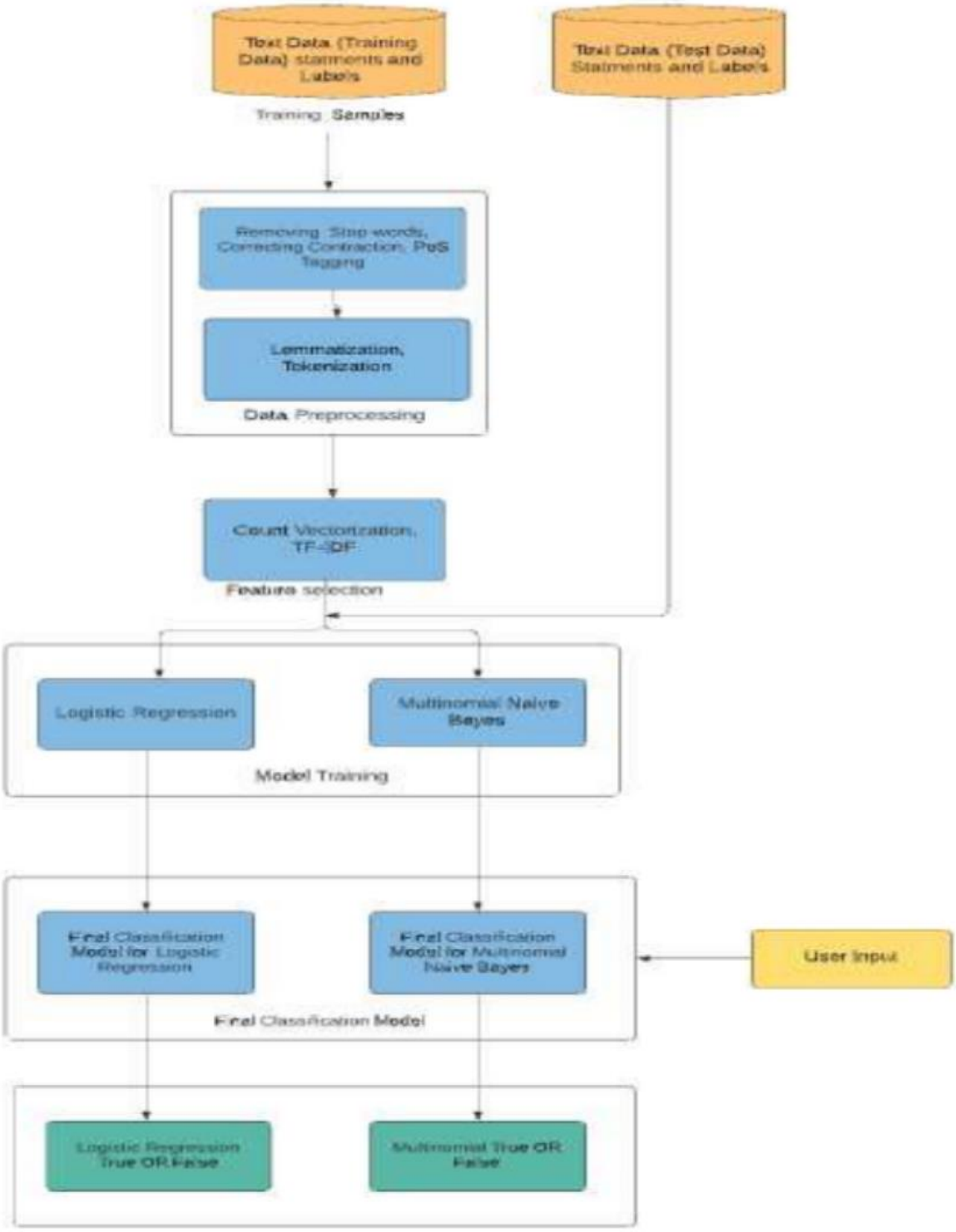


figure 5.1.1 Flowchart

5.1.2 Data Flow Diagram

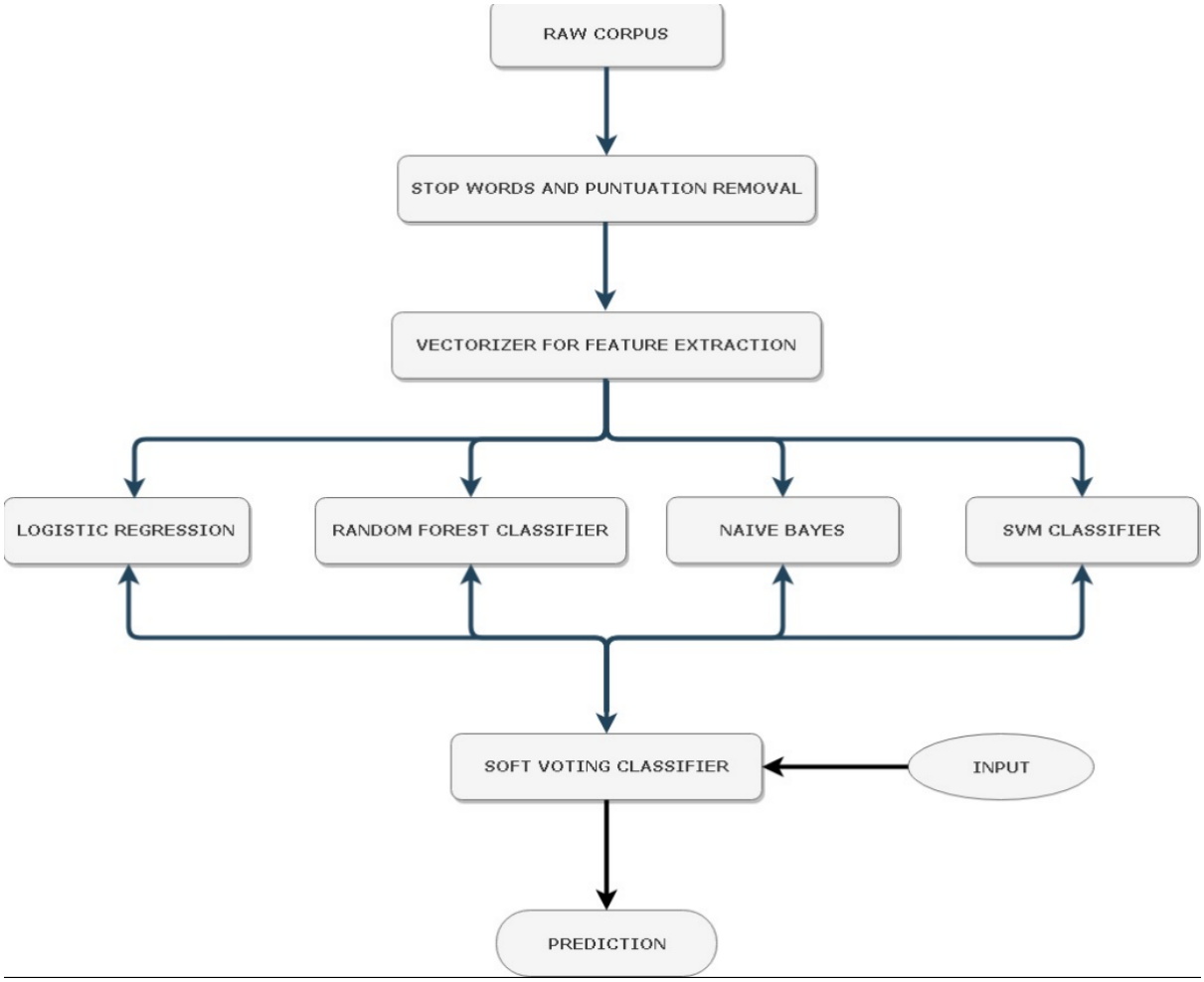
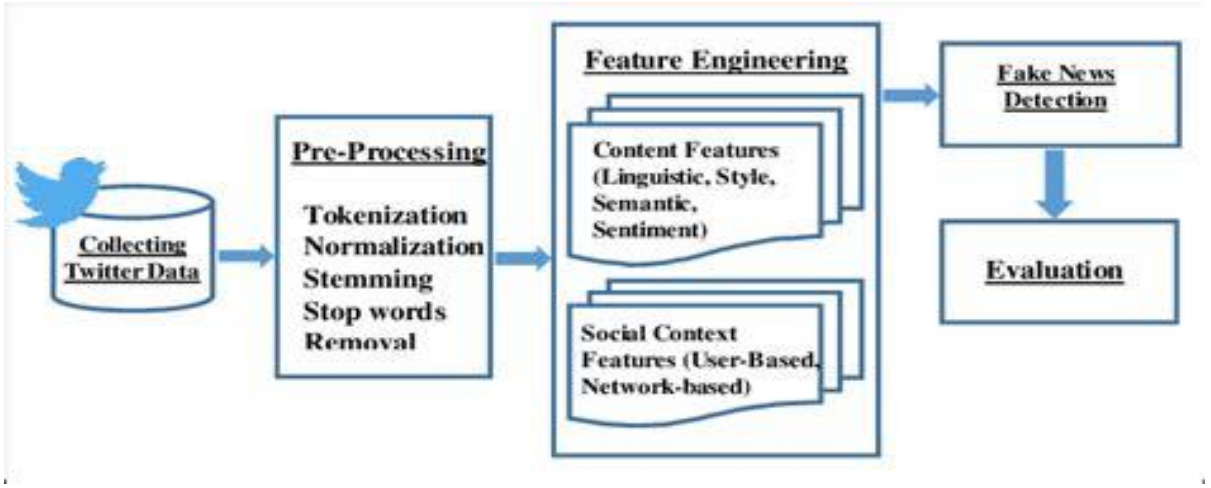


Figure 5.1.2 Data Flow Diagram

5.1.3 Class Diagram



5.1.3 Class Diagram

UNIT 6:

CODE IMPLEMENTATION

6.1 Screenshots

6.1.1 Work Flow

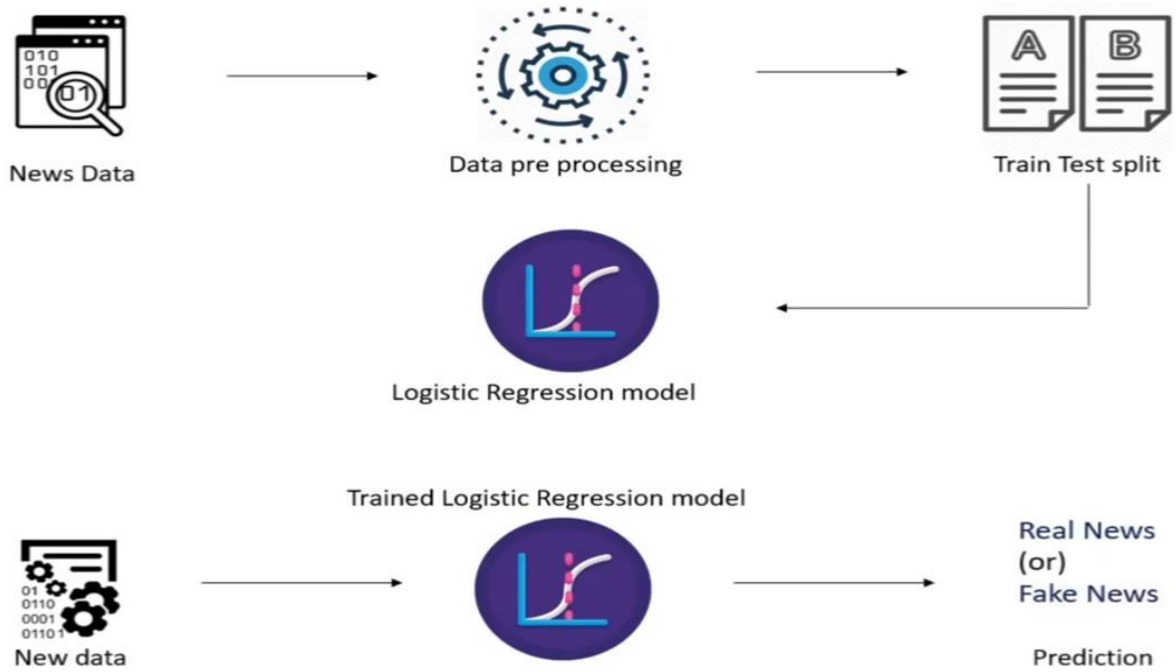


Figure 6.1.1 Work Flow

6.1.2 Libraries Used

Importing the Dependencies

```
1 import numpy as np
2 import pandas as pd
3 import re
4 from nltk.corpus import stopwords
5 from nltk.stem.porter import PorterStemmer
6 from sklearn.feature_extraction.text import TfidfVectorizer
7 from sklearn.model_selection import train_test_split
8 from sklearn.linear_model import LogisticRegression
9 from sklearn.metrics import accuracy_score
```

Figure 6.1.2 Libraries Used

6.1.3 Excel file to data frame

```
1 # print the first 5 rows of the dataframe
2 news_dataset.head()
```

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

Figure 6.1.3 Excel File To Data Frame

6.1.4 Logistic Regression

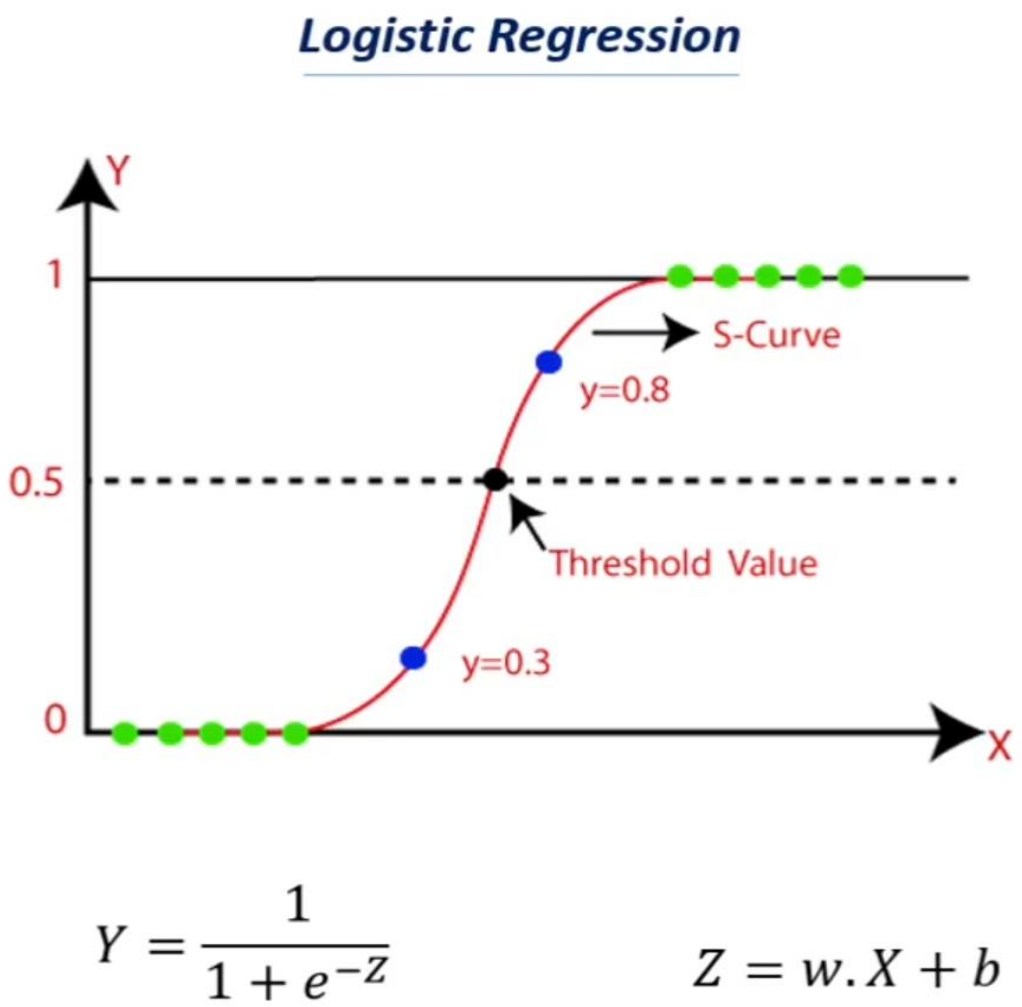


Figure 6.1.4 Logistic Regression

CHAPTER 7:
LIMITATIONS

Though we tried our best in developing this system but as limitations are mere parts of any system so are of our system. Some limitations of this system are: -

- No model is perfect so 100% accuracy can’t be reached
- Accuracy of this system depends on the size and quality of database used to train the model

CHAPTER 8:
CONCLUSION

In this , we've used Logistic Regression and Multinomial Naïve Bayes classifier which will predict the truthfulness of user input news, here we have presented a prediction model with feature selection used as Count Vectorization, TF-IDF which helps the model to be more accurate. We have investigated different classifier model with feature extraction as Count Vectorization, TF-IDF. The proposed Logistic Regression model achieves the mean accuracy of 0.93 with alpha value set as 0.6.