

# **SCRUTINIZING FAKE NEWS ON SOCIAL MEDIA USING MACHINE LEARNING TECHNIQUES**

Submitted in partial fulfillment of the requirements for  
the award of

Bachelor of Technology Degree in Information Technology

By

**BATHULA SOWMYA REDDY (37120013)**  
**JENILA J (37120035)**



**DEPARTMENT OF INFORMATION TECHNOLOGY**

**SCHOOL OF COMPUTING**

**SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,  
CHENNAI – 600119. TAMIL NADU**

**MARCH 2021**



**DEPARTMENT OF INFORMATION TECHNOLOGY**  
**BONAFIDE CERTIFICATE**

This is to certify that this Professional Training Report is the bonafide work of Bathula Sowmya Reddy (Reg no:37120013), JENILA J (Reg. No. 37120035) who carried out the project entitled "**SCRUTINIZING FAKE NEWS ON SOCIAL MEDIA USING MACHINE LEARNING TECHNIQUES**" under my supervision from November 2020 to April 2021.

**Internal Guide**

**DR.S. GOWRI, M.E., Ph.D.,**

**Head of the Department**

**Dr. R.SUBHASHINI, M.E.,Ph.D.,**

---

**Submitted for Viva voce Examination held on \_\_\_\_\_**

**Internal Examiner**

**External Examiner**

## **DECLARATION**

I, **BATHULA SOWMYA REDDY (37120013), JENILA J (37120035)** hereby declare that the Professional Training Report on “**SCRUTINIZING FAKE NEWS ON SOCIAL MEDIA USING MACHINE LEARNING TECHNIQUES**” done by me under the guidance of guide **Dr.S.GOWRI M.E.,Ph.D.**, Department of Information Technology at Sathyabama Institute of Science and technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Information Technology.

**DATE:**

**PLACE:**

**SIGNATURE OF THE CANDIDATE**

## **ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.Sasikala M.E.,Ph.d, Dean**, School of Computing, **Dr. Subashini M.E.,Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S.Gowri M.E.,Ph.D.**, for this valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

## ABSTRACT

Detection of fake news online is important in today's society as fresh news content is rapidly being produced as a result of the abundance of technology that is present. In the world of false news, there are seven main categories and within each category, the piece of fake news content can be visual- and/or linguistic-based. In order to detect fake news, both linguistic and non-linguistic cues can be analyzed using several methods. While many of these methods of detecting fake news are generally successful, they do have some limitations. Recently, there has been a sudden increase in fake news due to the immense use of social media and online news media. It has become much easier to spread fake news than earlier. This type of fake news when spread may cause a severe impact. Hence it is very essential to take certain measures to reduce or differentiate between real and fake news.

**TF (Term Frequency)** The number of times a word appears in a document is its Term Frequency. A higher value means a term appears more often than others, and so, the document is a good match when the term is part of the search terms.

**IDF (Inverse Document Frequency)** Words that occur many times a document, but also occur many times in many others, may be irrelevant. IDF is a measure of how significant a term is in the entire corpus. In this paper, we presents an analysis of fake news detection using various classifiers like Passive aggressive Classifier, Naïve Bayes, random forest, Support Vector Machine (SVM), stochastic gradient descent, Logistic regression and decision tree classifier. we get the accuracy and compared result.

## TABLE OF CONTENT

| CH NO    | TITLE NAME                                 | PAGE NO   |
|----------|--------------------------------------------|-----------|
|          | ABSTRACT                                   | I         |
|          | LIST OF FIGURES                            | V         |
|          | LIST OF TABLE                              | VI        |
|          | LIST OF ABBRIEVATION                       | VII       |
| <b>1</b> | <b>INTRODUCTION</b>                        | <b>1</b>  |
|          | 1.1 DOMAIN DESCRIPTION                     | 1         |
|          | 1.2 INTRODUCTION OF THE PROJECT            | 2         |
|          | 1.3 MOTIVATION                             | 3         |
|          | 1.4 OBJECTIVE                              | 4         |
| <b>2</b> | <b>LITERATURE SURVEY</b>                   | <b>5</b>  |
|          | 2.1 EXISTING SYSTEM                        | 5         |
| <b>3</b> | <b>EXPERIMENTAL METHODS AND ALGORITHMS</b> | <b>10</b> |
|          | 3.1 HARDWARE REQUIREMENTS                  | 10        |
|          | 3.2 SOFTWARE REQUIREMENTS                  | 10        |
|          | 3.3 PROPOSED SYSTEM                        | 10        |
|          | 3.3.1 ADVANTAGES                           | 11        |
|          | 3.4 FEASIBILITY SYTEM                      | 11        |
|          | 3.4.1 ECONOMICAL FEASIBILITY               | 11        |
|          | 3.4.2 TECHNICAL FEASIBILITY                | 11        |
|          | 3.4.3 SOCIAL FEASIBILITY                   | 12        |
|          | 3.5 LANGUAGE DESCRIPTION                   | 12        |
|          | 3.5.1 ABOUT PYTHON LANGUAGE                | 12        |

|                                          |    |
|------------------------------------------|----|
| 3.5.2 PYTHON PROGRAMMING CHARACTERISTICS | 12 |
| 3.5.3 APPLICATIONS OF PYTHON PROGRAMMING | 13 |
| 3.6 FEATURES OF ANACONDA NAVIGATOR       | 15 |
| 3.6.1 ANACONDA                           | 15 |
| 3.6.2 APPLICATIONS PROVIDED IN ANACONDA  | 16 |
| 3.7 NEW FEATURES OF ANACONDA 5.3         | 18 |
| 3.7.1 OBJECTIVES                         | 19 |
| 3.8 SYSTEM DESIGN                        | 19 |
| 3.8.1 SYSTEM ARCHITECTURE                | 19 |
| 3.9 UML DIAGRAM                          | 20 |
| 3.9.1 GOALS                              | 20 |
| 3.10 USE CASE DIAGRAM                    | 21 |
| 3.11 SEQUENCE DIAGRAM                    | 21 |
| 3.12 ACTIVITY DIAGRAM                    | 22 |
| 3.13 METHODOLOGIES                       | 23 |
| 3.13.1 DATA COLLECTION                   | 23 |
| 3.13.2 DATA PREPROCESSING                | 23 |
| 3.13.3 FEATURE EXTRACTION                | 24 |
| 3.13.4 MODEL CREATION                    | 24 |
| 3.13.5 CLASSIFICATION                    | 24 |
| 3.13.5.1 ALGORITHMS                      | 24 |
| 3.14 EVALUATION METRICS                  | 28 |
| 3.15 TESTING                             | 28 |
| 3.15.1 SOFTWARE TESTING                  | 28 |

|          |                                   |           |
|----------|-----------------------------------|-----------|
|          | 3.15.2 TEST CASE                  | 29        |
|          | 3.15.3 TESTING TECHNIQUES         | 29        |
| <b>4</b> | <b>RESULT AND DISCUSSION</b>      | <b>32</b> |
|          | 4.1 RESULTS                       | 32        |
|          | 4.2 COMPARISON GRAPH              | 35        |
| <b>5</b> | <b>CONCLUSION AND ENHANCEMENT</b> | <b>36</b> |
|          | 5.1 CONCLUSION                    | 36        |
|          | 5.2 FUTURE WORKS                  | 36        |
|          | REFERENCES                        | 37        |
|          | <b>APPENDICES</b>                 | <b>39</b> |
|          | A. SOURCE CODE                    | 39        |
|          | B. SCREENSHOTS                    | 46        |
|          | C. PAPER PUBLICATION              | 51        |

## LIST OF FIGURES

| FIGURE NO | FIGURE NAME                        |
|-----------|------------------------------------|
| 3.1       | Anaconda Distribution              |
| 3.2       | Anaconda Distribution Applications |
| 3.3       | System Architecture                |
| 3.4       | Use-Case Diagram                   |
| 3.5       | Sequence Diagram                   |
| 3.6       | Activity Diagram                   |
| 4.1       | Output for PA                      |
| 4.2       | Output for LR                      |
| 4.3       | Output for DT                      |
| 4.4       | Output for RFC                     |
| 4.5       | Output for NB                      |
| 4.6       | Output for SVM                     |
| 4.7       | Output for SGDC                    |
| 4.8       | Overall Accuracy graph 1           |
| 4.9       | Overall Accuracy graph 2           |

## LIST OF TABLE

| TABLE NO  | TABLE NAME       |
|-----------|------------------|
| Table 4.1 | Overall accuracy |

## **LIST OF ABBREIVATION**

|               |                                           |
|---------------|-------------------------------------------|
| <b>PA</b>     | Passive Aggressive                        |
| <b>LR</b>     | Logistic Regression                       |
| <b>DTC</b>    | Decision Tree Classifier                  |
| <b>RFC</b>    | Random Forest Classifier                  |
| <b>MNB</b>    | Multi Nominal Naïve Bayes                 |
| <b>SVM</b>    | Support Vector Machine                    |
| <b>SGDC</b>   | Stochastic Gradient Descent Classifier    |
| <b>TF-IDF</b> | Team Frequency Inverse Document Frequency |
| <b>AI</b>     | Artificial Intelligence                   |
| <b>RNN</b>    | Recurrent Neural Network                  |
| <b>IDE</b>    | Integrated Development Environment        |
| <b>CMS</b>    | Content Management System                 |
| <b>API</b>    | Application Programming Interface         |
| <b>UML</b>    | Unified Modeling Language                 |

# CHAPTER 1

## INTRODUCTION

### 1.1 Domain description

#### Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

#### Machine Learning Methods

Machine learning algorithms are often categorized as supervised or unsupervised.

**Supervised machine learning algorithms** can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

In contrast, **unsupervised machine learning algorithms** are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn't figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.

**Semi-supervised machine learning algorithms** fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of

unlabeled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiring unlabeled data generally doesn't require additional resources.

**Reinforcement machine learning algorithms** is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal.

### **Advantages of Machine Learning**

Machine Learning undoubtedly helps people to work more creatively and efficiently. Basically, you too can delegate quite complex or monotonous work to the computer through Machine Learning - starting with scanning, saving and filing paper documents such as invoices up to organizing and editing images.

In addition to these rather simple tasks, self-learning machines can also perform complex tasks. These include, for example, the recognition of error patterns. This is a major advantage, especially in areas such as the manufacturing industry: the industry relies on continuous and error-free production. While even experts often cannot be sure where and by which correlation a production error in a plant fleet arises, Machine Learning offers the possibility to identify the error early this saves down times and money. Self-learning programs are now also used in the medical field. In the future, after "consuming" huge amounts of data (medical publications, studies, etc.), apps will be able to warn a in case his doctor wants to prescribe a drug that he cannot tolerate. This "knowledge" also means that the app can propose alternative options which for example also take into account the genetic requirements of the respective patient.

### **1.2 Introduction of the project:**

Traditionally people got news from various trusted sources, but nowadays people get news from online sources and social media which now has become a trend,

and most of the time it becomes difficult to decide whether stories are authentic or not. Social media sites can have a major influence in expanding fake news.

Fake news is a news created to intentionally misguide or mislead readers. Ever since the birth of social media and online news media, the spread of fake news has increased drastically. Social media sites such as Facebook, Whatsapp, twitter etc. are one of the biggest sources of spreading fake news. This paper provides a survey on recent past research papers done on this particular region and also proposes a model that can predict whether the news is real or fake using NLP techniques and Machine Learning. The model will be using the URL as an input that not only validates the headline but also validates the behavior of the site and other related parameters that can provide an improvement over past research. For detection whether the news is fake or not we are implementing a fake news detection system and comparing the accuracy of the system using various machine learning models.

### **1.3 Motivation**

The widespread problem of fake news is very difficult to tackle in today's digital world where there are thousands of information sharing platforms through which fake news or misinformation may propagate. It has become a greater issue because of the advancements in AI which brings along artificial bots that may be used to create and spread fake news . The situation is dire because many people believe anything they read on the internet and the ones who are amateur or are new to the digital technology may be easily fooled. A similar problem is fraud that may happen due to spam or malicious emails and messages. So, it is compelling enough acknowledge this problem take on this challenge to control the rates of crime, political unrest, grief, and thwart the attempts of spreading fake news.

## **1.4 OBJECTIVES**

The main objective is to detect the fake news, which is a classic text classification problem with a straight forward proposition. It is needed to build a model that can differentiate between “Real” news and “Fake” news

## **CHAPTER -2**

### **LITERATURE SURVEY**

#### **2.1 EXISTING SYSTEM**

Fake news has been gathering a lot of attention worldwide recently. The effects can be political, economic, organizational, or even personal. In Existing paper discusses the approach of natural language processing and machine learning in order to solve this problem. Use of bag-of-words, n-grams, count vectorizer has been made, TF-IDF, and trained the data on three classifiers to investigate which of them works well for this specific dataset of labeled news statements. Passive aggressive Classifier, Random forest, Support Vector Machine.

##### **2.1.1 Disadvantages:**

- Each algorithm are well trained and only one gets accuracy
- Others get low accuracy

**1.Title: “CSI: A Hybrid Deep Model for Fake News Detection, ”CIKM17 , Singapore, Singapore.Author:Natali Ruchansky, Sungyong Seo and Yan Liu Year:2017**

Presented a model which is spurred by the three qualities of fake news for example the content of an article, the client reaction it gets, and the source client advancing it. In light of these three qualities display is made out of three modules: Capture, Score, and Integrate. Capture depends on the reaction and content. To remove transient portrayal of articles Recurrent Neural Network (RNN) is used. Score module learns the source qualities dependent on the conduct of clients, and these two modules incorporated with third module to group an article as fake or genuine.

**2. Title:“Automatically Identifying Fake News in Popular Twitter Threads,” 2017 IEEE International Conference on Smart Cloud. Author:Cody Buntain and Jennifer Golbeck Year:2017**

Developed a method for fake news recognition on Twitter by figuring out how to anticipate exactness assessment in two believably centered Twitter dataset: CREDBANK and PHEME. CREDBANK is a publicly supported dataset of precision assessment for occasions in Twitter. PHEME is a dataset of potential talk in Twitter

and journalistic evaluation of their exactness. This technique is then connected to Twitter content sourced from Buzz feed's fake news dataset. An element examination at that point identities highlights that are most prescient for publicly supported and journalistic precision evaluations, aftereffects of which are steady with earlier work.

**3. Title: "Spotting Fake News: A Social Argumentation Framework for Scrutinizing Alternative Facts," 2017 IEEE 24th International Conference on Web Services. Author: Ricky J. Sethi Year: 2017**

System is countering the spread of fake news on social networks by leveraging these crowds to instead help verify alternative facts. A prototype of social argumentation framework to verify the validity of proposed alternative facts to help curb the propagation of fake news. It utilizes fundamental argumentation ideas in a graph-theoretic framework that also incorporates semantic web and linked data principles. The argumentation structure is crowdsourced and mediated by expert moderators in a virtual community. It consists of three main components: The Graph-Theoretic Framework, the User Interface Component, and the Virtual Community to support crowdsourcing.

**4. Title: "Automatic Online Fake News Detection Combining Content and Social Signals," PROCEEDING OF THE 22ND CONFERENCE OF FRUCT ASSOCIATION. Author: Marco L. Della Vedova, Eugenio Tacchini, Ballarin Year: 2018**

Proposed a novel ML fake news detection method which, by combining news content and social context features, outperforms existing methods in the literature, increasing their already high accuracy by up to 4.8%. Second, they implemented method within a Facebook Messenger Chat bot and validate it with a real-world application, obtaining a fake news detection accuracy of 81.7%

**5. Title: "CIMTDetect: A Community Infused Matrix-Tensor Coupled Factorization Based Method for Fake News Detection," 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). Author: Shashan Gupta, Raghuveer Thirukovalluru Year: 2018**

The authors describe a system in which they exploit the echo chamber communities i.e. the communities that share the same belief that exist in the social media. Because the spreading of fake news on echo chambers is easy for this they use 3-mode tensor representation of the news - <News, User, Community>. It is a novel state of art technique.

**6. Title: "Identifying Tweets With Fake News," 2018 IEEE International Conference on Information Reuse and Integration for Data Science.**

**Author: Saranya Krishnan and Min Chen Year: 2018**

Uses advance framework for fake news detection including techniques like statistical analysis of twitter user accounts, reverse image searching, cross verification of fake news sources and data mining. The framework contains two components core and website. Core is responsible for fetching the tweets content, constructing the feature set using the tweet dataset provided by the dataset, building the classifier and generating the evaluation report. website is to provide tweet credibility predictions, tweet details and crowd sourcing results.

**7. Title: "Understanding User Profiles on Social Media for Fake News Detection," 2018 IEEE Conference on Multimedia Information Processing and Retrieval. Author: Kai Shu, Suhag Wang and Huan Liu Year: 2018**

The performance of detecting fake news only from content is not satisfactory, and it is suggested to incorporate user social engagements to improve fake news detection. It will construct a real-world datasets measuring users trust level on the fake news. There are two type of users" Experienced" users who can recognize fake news and" Naïve" who can believe fake news. A comparative analysis over implicit and explicit profile features between these users which reveals their potential to differentiate fake news.

**8. Title: "Hybrid Machine-Crowd Approach for Fake News Detection," 2018 IEEE 4th International Conference on Collaboration and Internet Computing. Author: Shaban Shabani and Maria Sokhn Year: 2018**

Addresses the fake news and satire detection by proposing a method that uses a hybrid machine-crowd approach for detection of potentially deceptive news. This system combines the human factor with the machine learning approach and a

decision-making model that estimates the classification confidence of algorithms and decides whether the task needs human input or not. This approach achieves reasonably higher accuracy compared other results, in exchange of cost and latency of using the crowdsourcing service. Logistic Regression (Log), SVM, Random Forest (RF), Neural Networks (NN), and Gradient Boosting Classifier (GDC) Machine learning classifications models are used. While crowdsourcing works on the judgments of online cloud works.

**9. Title:“Fake News Detection Enhancement with Data Imputation,” 2018 IEEE 16th Int. Conf. on Dependable, Autonomic Secure Comp., 16th Int. Conf. on Pervasive Intelligence Comp., 4th Int. Conf. on Big Data Intelligence Comp., and 3rd Cyber Sci. Tech. Cong. Author:Chandra Mouli Madhav Kotteti, Xishuang Dong, Na Li and Lijun Qian Year:2018**

The authors describe a system in which they use machine learning classification models combined with the proposed data preprocessing methods. They have used data imputation for handling of categorical and missing values. Used TF-IDF vectorization for feature extraction to filter out irrelevant features. Used Multi-Layer Perceptron (MLP) classifier and the accuracy is increased by 15%.

**10. Title:“Weakly Supervised Learning for Fake News Detection on Twitter,” 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). Author:Stefan Helmstetter and Heiko Paulheim Year:2018**

It uses a weakly supervised approach, which will collect a large amount of noisy dataset from many tweets. During collection of the data, tweets are classified to trustworthy and untrustworthy source and train classifier on these datasets. Then the system uses these classifiers for the detection of fake news i.e. fake tweets. To improve the classification of trustworthy and untrustworthy sources a factor F1 is used up to 0.9.

**11. Akshay Jain, Amey Kasbe, “ Fake News Detection”, 2018 IEEE International conference for student, Electrical and Electronics and computer science**

This paper describes a simple fake news detection method based on one of the machine learning algorithms, detecting fake news from four different

algorithm and comparing the accuracy LSTM acquires 94%, among other algorithm – NaïveBayes classifier, etc.,

**12. Ehesas Mia Mahir, Abdullah-All-Tanvir,” Detecting Fake News using Machine Learning and Deep Learning Algorithm”, 2019 7th International Conference on Smart Computing**

In this paper social media spreads fake news with great source and that can be compared with three well-known algorithm SVM, Naïve Bayes, and logical regression etc.,

**13. M. Granik and V. Mesyura, “Fake news detection using Naïve Bayes classifier,” 2017 IEEE 1st Ukr. Conf. Electr. Comput. Eng. UKRCON 2017 - Proc.,**

This paper proposed an approach for detection of fake news using Naïve Bayes classifier with accuracy of 74% on the test set.

**14. N. Smitha; R. Bharath” Performance Comparison of Machine Learning Classifiers for Fake News Detection” 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA) Year: 2020**

In this paper the false information is create deliberate situation and unexpected engendered over social media. It compares performance of many different classifier with three different modules, count vectorizer, TF-IDF vectorizer and word embedding. As a result neural network comparing all classifications pulls off the best accuracy.

**15. Rahul R Mandical;N Mamatha;N Shivakumar;R Monica;A N Krishna “Identification of Fake News Using Machine Learning” 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) Year: 2020**

In this paper, three different machine learning classifiers like NaïveBayes, PassiveAggressive, and deep neural networks are used and show that among the proposed neural network outperformed both NaïveBayes and PassiveAggressive

## **CHAPTER 3**

## **EXPERIMENTAL METHODS AND ALGORITHMS**

### **3.1 HARDWARE REQUIREMENTS**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list, especially in case of operating systems. The minimal hardware requirements are as follows,

1. PROCESSOR : PENTIUM IV
2. RAM : 8 GB
3. PROCESSOR : 2.4 GHZ
4. MAIN MEMORY : 8GB RAM
5. PROCESSING SPEED : 600 MHZ
6. HARD DISK DRIVE : 1TB
7. KEYBOARD :104 KEYS

### **3.2 SOFTWARE REQUIREMENTS**

Software requirements deals with defining resource requirements and prerequisites that needs to be installed on a computer to provide functioning of an application. These requirements are need to be installed separately before the software is installed. The minimal software requirements are as follows,

1. FRONT END :PYTHON
2. IDE : ANACONDA
3. OPERATING SYSTEM :WINDOWS 10

### **3.3 PROPOSED SYSTEM**

This paper discusses the approach of machine learning in order to solve this problem. Use of count vectorizer , TF-IDF, and trained the data on seven classifiers to investigate which of them works well for this specific dataset of labelled news statements. we presents an analysis of fake news classification using various classifiers like Passive aggressive Classifier,Naïve Bayes, Random forest,Support Vector Machine (SVM),stochastic gradient descent, Logistic regression and

decision tree classifier. we get the accuracy and compared result. The precision, recall and f1 scores help us determine which model works best.

### **3.3.1 Advantages:**

- Each algorithm are well trained and good accuracy
- Sgdc and passive aggressive classifier gets high accuracy
- Sgdc-93% and passive aggressive classifier - 92%

## **3.4 FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are

1. Economic feasibility
2. Technical feasibility
3. Social feasibility

### **3.4.1 ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### **3.4.2 TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. A feasibility study evaluates the project's potential for success.

### **3.4.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

## **3.5 LANGUAGE DESCRIPTION**

### **3.5.1 About Python Language**

Python is an object-oriented programming language created by Guido Rossum in 1989. It is ideally designed for rapid prototyping of complex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies use the Python programming language include NASA, Google, YouTube, BitTorrent, etc. Python programming is widely used in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced fields of Computer Science. Python had deep focus on code readability & this class will teach you python from basics.

### **3.5.2 Python Programming Characteristics**

- It provides rich data types and easier to read syntax than any other programming languages
- It is a platform independent scripted language with full access to operating system API
- Compared to other programming languages, it allows more run-time flexibility
- It includes the basic text manipulation facilities of Perl and Awk
- A module in Python may have one or more classes and free functions
- Libraries in Python are cross-platform compatible with Linux, Macintosh, and Windows
- For building large applications, Python can be compiled to byte-code
- Python supports functional and structured programming as well as OOP
- It supports interactive mode that allows interacting Testing and debugging of snippets of code

- In Python, since there is no compilation step, editing, debugging and testing is fast.

### **3.5.3 Applications of Python Programming**

#### **Web Applications**

You can create scalable Web Apps using frameworks and CMS (Content Management System) that are built on Python. Some of the popular platforms for creating Web Apps are: Django, Flask, Pyramid, Plone, Django CMS. Sites like Mozilla, Reddit, Instagram and PBS are written in Python.

#### **Scientific and Numeric Computing**

There are numerous libraries available in Python for scientific and numeric computing. There are libraries like: SciPy and NumPy that are used in general purpose computing. And, there are specific libraries like: EarthPy for earth science, AstroPy for Astronomy and so on. Also, the language is heavily used in machine learning, data mining and deep learning.

#### **Creating software Prototypes**

Python is slow compared to compiled languages like C++ and Java. It might not be a good choice if resources are limited and efficiency is a must. However, Python is a great language for creating prototypes. For example: You can use Pygame (library for creating games) to create your game's prototype first. If you like the prototype, you can use language like C++ to create the actual game.

#### **Good Language to Teach Programming**

Python is used by many companies to teach programming to kids and newbies. It is a good language with a lot of features and capabilities. Yet, it's one of the easiest language to learn because of its simple easy-to-use syntax.

#### **Python GUI Tkinter**

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

To create a tkinter app:

- ◆ Importing the module – tkinter
- ◆ Create the main window (container)

- ◆ Add any number of widgets to the main window
- ◆ Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code. Note that the name of the module in Python 2.x is 'tkinter' and in Python 3.x it is 'tkinter'. 'tkinter' also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes class.

- ✓ pack() method:It organizes the widgets in blocks before placing in the parent widget.
- ✓ grid() method:It organizes the widgets in grid (table-like structure) before placing in the parent widget.
- ✓ place() method:It organizes the widgets by placing them on specific positions directed by the programmer.

### **About OpenCV Package**

Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability.

Compared to other languages like C/C++, Python is slower. But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. This gives us two advantages: first, our code is as fast as original C/C++ code (since it is the actual C++ code working in background) and second, it is very easy to code in Python. This is how OpenCV-Python works, it is a Python wrapper around original C++ implementation.

And the support of Numpy makes the task more easier. Numpy is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which increases number of weapons in your arsenal. Besides that, several other libraries like SciPy, Matplotlib which supports Numpy can be used with this.

So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

## 3.6 FEATURES OF ANACONDA NAVIGATOR

### 3.6.1 Anaconda

Anaconda is a free and open source, easy to install distribution of Python and R programming languages. Anaconda provides a working environment which is used for scientific computing, data science, statistical analysis and machine learning.

The latest distribution of Anaconda is Anaconda 5.3 and is released in October, 2018. It has the conda package, environment manager and a collection of 1000+ open source packages long with free community support.

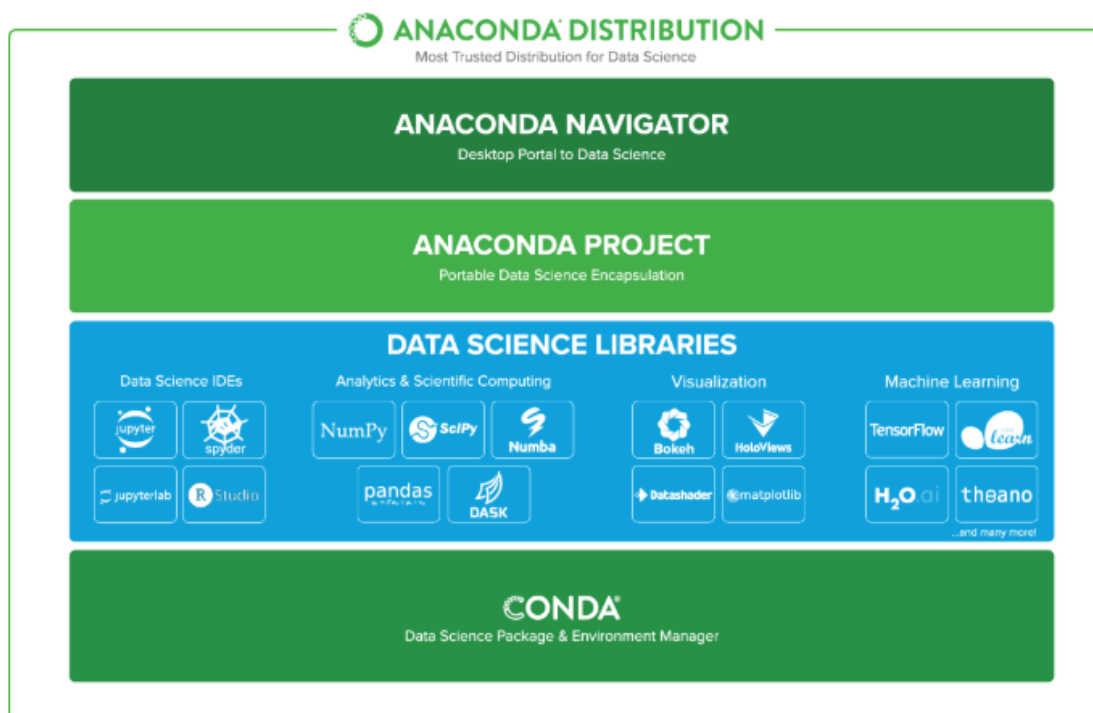


Fig: 3.1 Anaconda Distribution

### What is Anaconda Navigator?

Anaconda Navigator is a desktop graphical user interface (GUI) included in the Anaconda distribution. It allows us to launch applications provided in the Anaconda distribution and easily manage conda packages, environments and channels without the use of command-line commands. It is available for Windows, macOS and Linux.

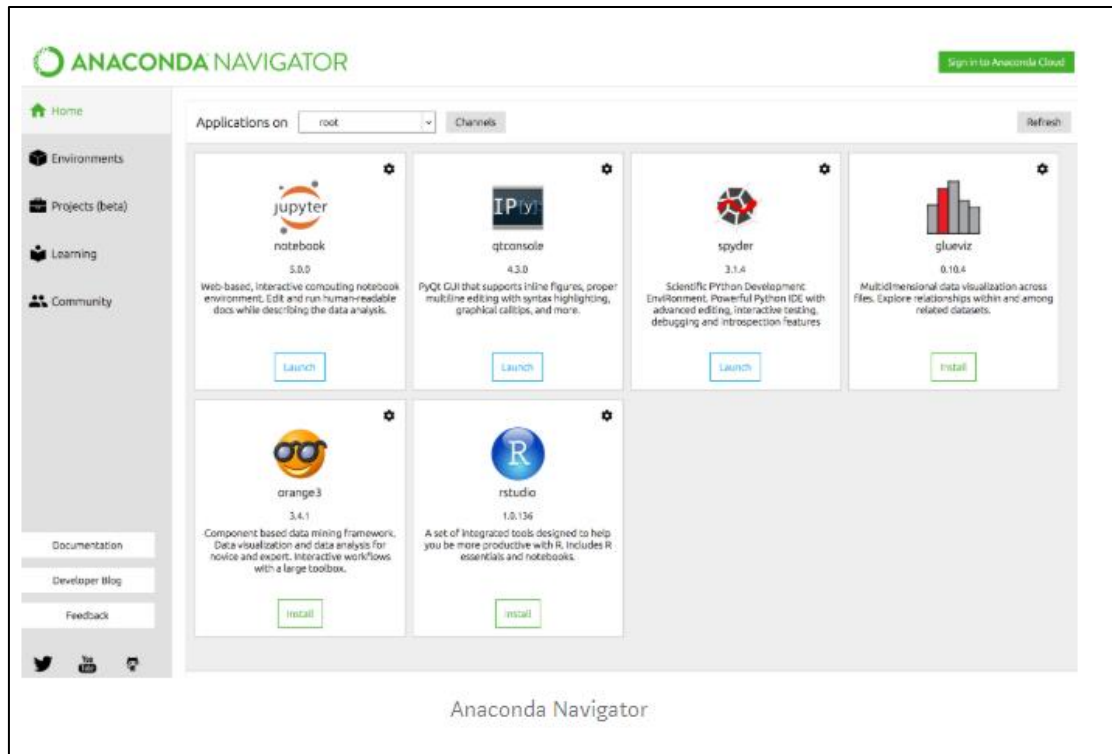


Fig:3.2 Anaconda Distribution Applications

### 3.6.2 Applications Provided In Anaconda Distribution

The Anaconda distribution comes with the following applications along with Anaconda Navigator.

1. JupyterLab
2. Jupyter Notebook
3. Qt Console
4. Spyder
5. Glueviz
6. Orange3
7. RStudio

## 8. Visual Studio Code

> **JupyterLab:** This is an extensible working environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

> **Jupyter Notebook:** This is a web-based, interactive computing notebook environment. We can edit and run human-readable docs while describing the data analysis.

> **Qt Console:** It is the PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips and more.

> **Spyder:** Spyder is a scientific Python Development Environment. It is a powerful Python IDE with advanced editing, interactive testing, debugging and introspection features.

> **VS Code:** It is a streamlined code editor with support for development operations like debugging, task running and version control.

> **Glueviz:** This is used for multidimensional data visualization across files. It explores relationships within and among related datasets.

> **Orange 3:** It is a component-based data mining framework. This can be used for data visualization and data analysis. The workflows in Orange 3 are very interactive and provide a large toolbox.

> **Rstudio:** It is a set of integrated tools designed to help you be more productive with R. It includes R essentials and notebooks.

### 3.7 New Features of Anaconda 5.3



Compiled with Latest Python release: Anaconda 5.3 is compiled with Python 3.7, taking advantage of Python's speed and feature improvements.

- **Better Reliability:** The reliability of Anaconda has been improved in the latest release by capturing and storing the package metadata for installed packages.

- **Enhanced CPU Performance:** The Intel Math Kernel Library 2019 for Deep Neural Networks(MKL 2019) has been introduced in Anaconda 5.3 distribution. Users deploying Tensorflow can make use of MKL 2019 for Deep Neural Networks. These Python binary packages are provided to achieve high CPU performance.

- **New packages are added:** There are over 230 packages which has been updated and added in the new release.

- **Work in Progress:** There is a casting bug in Numpy with Python 3.7 but the team is currently working on patching it until Numpy is updated.

### 3.7.1 Objective

Our primary objective in doing this project was to build a device inspired from Leapmotion. It is a device which recognizes hand gestures and can be used to virtually control a computer. In short, it provides a virtual screen with which we can interact with the computer. But the required hardware for making a device on these lines was not feasible, in terms of budget and time frame provided. So, we decided to build an introductory software implementation of the device which would eventually act as a virtual mouse.

## 3.8 SYSTEM DESIGN

### 3.8.1 SYSTEM ARCHITECTURE

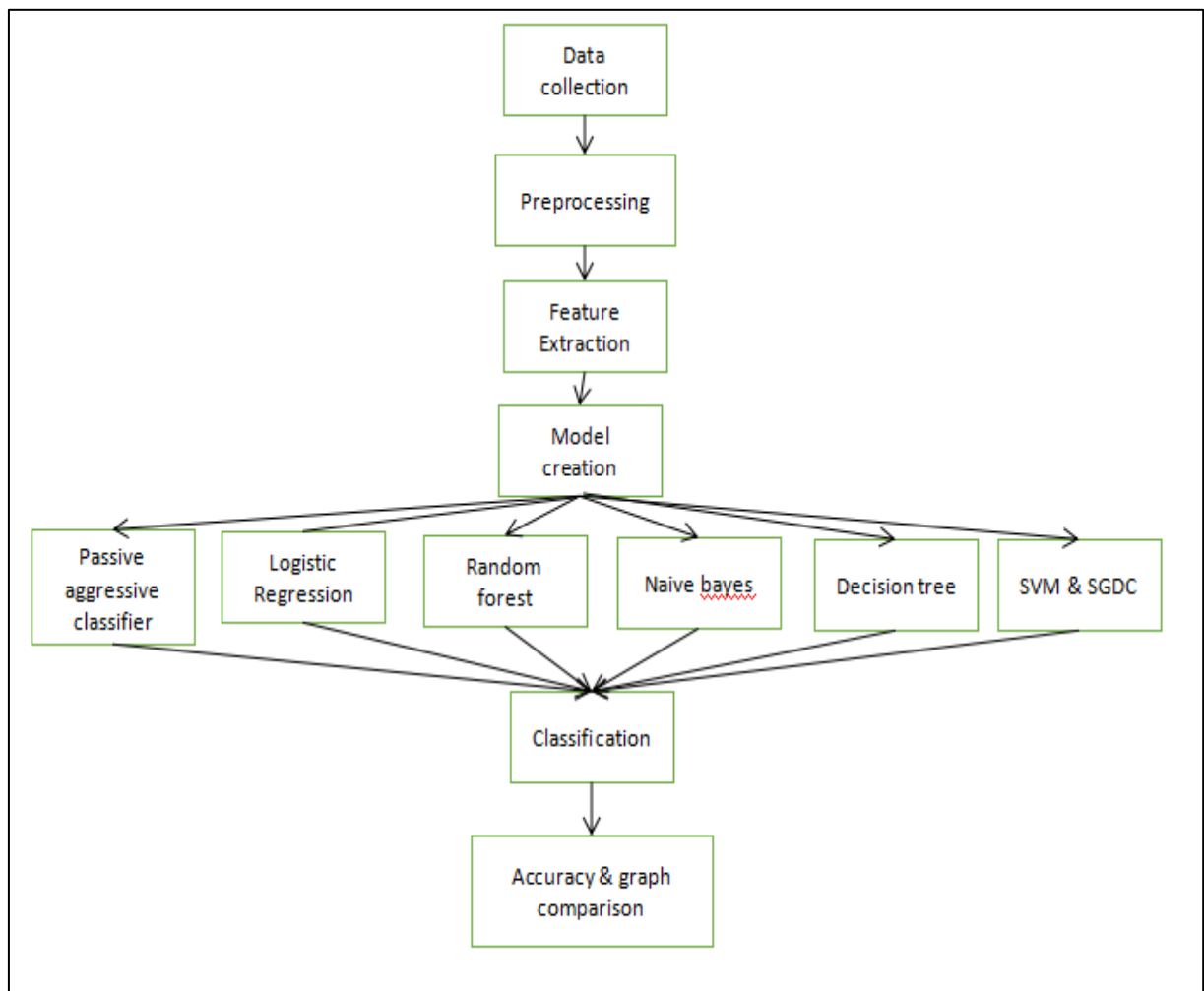


Fig:3.3 System Architecture

### **3.9 UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

- The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.
- The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.
- The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.
- The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

#### **3.9.1 GOALS**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extensibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

### 3.10 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

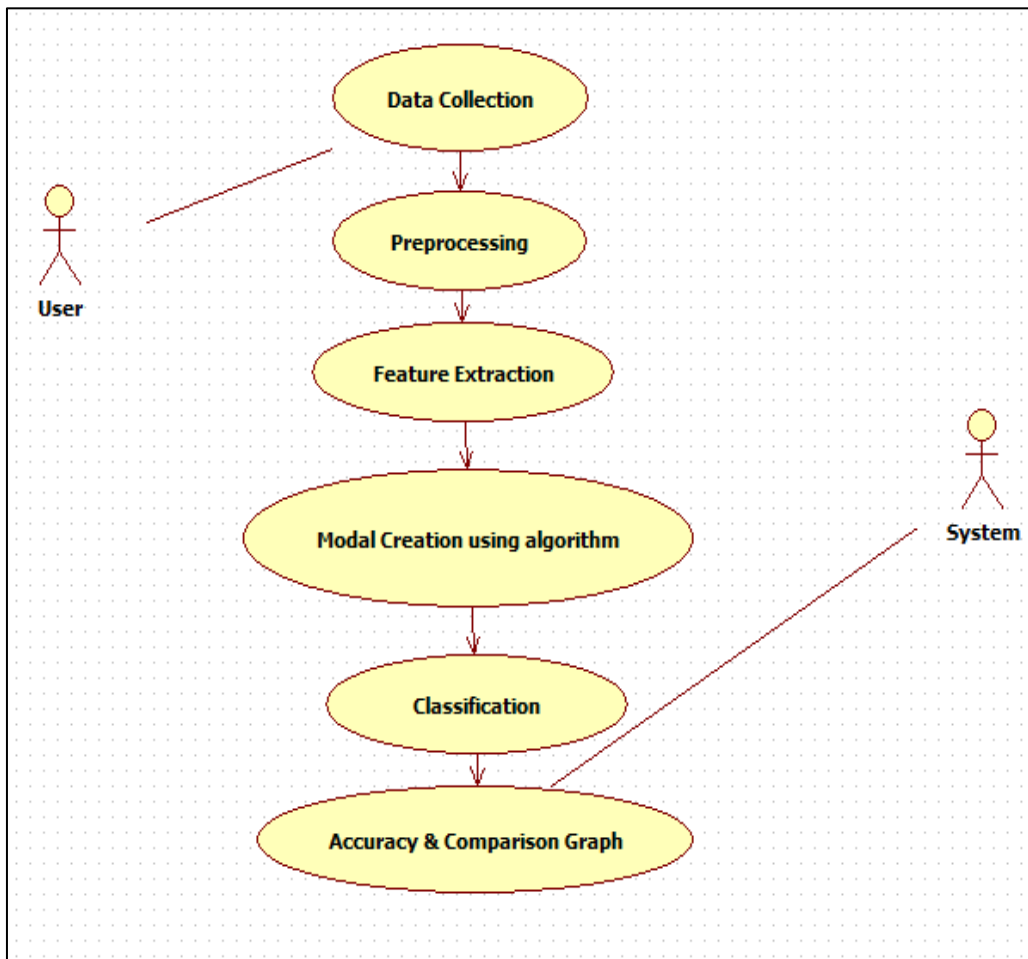


Fig:3.4 Use-Case Diagram for Scrutinizing Fake News on Social Media

### 3.11 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

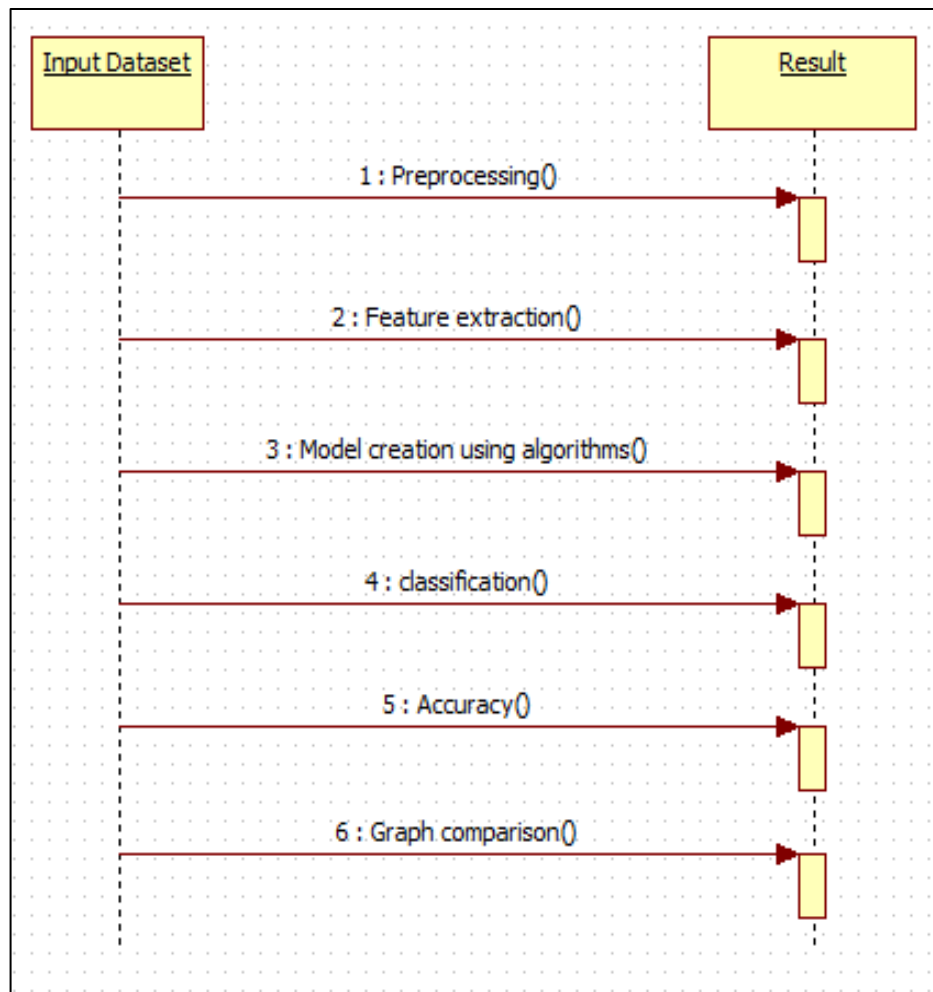


Fig: 3.5 Sequence Diagram

### 3.12 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

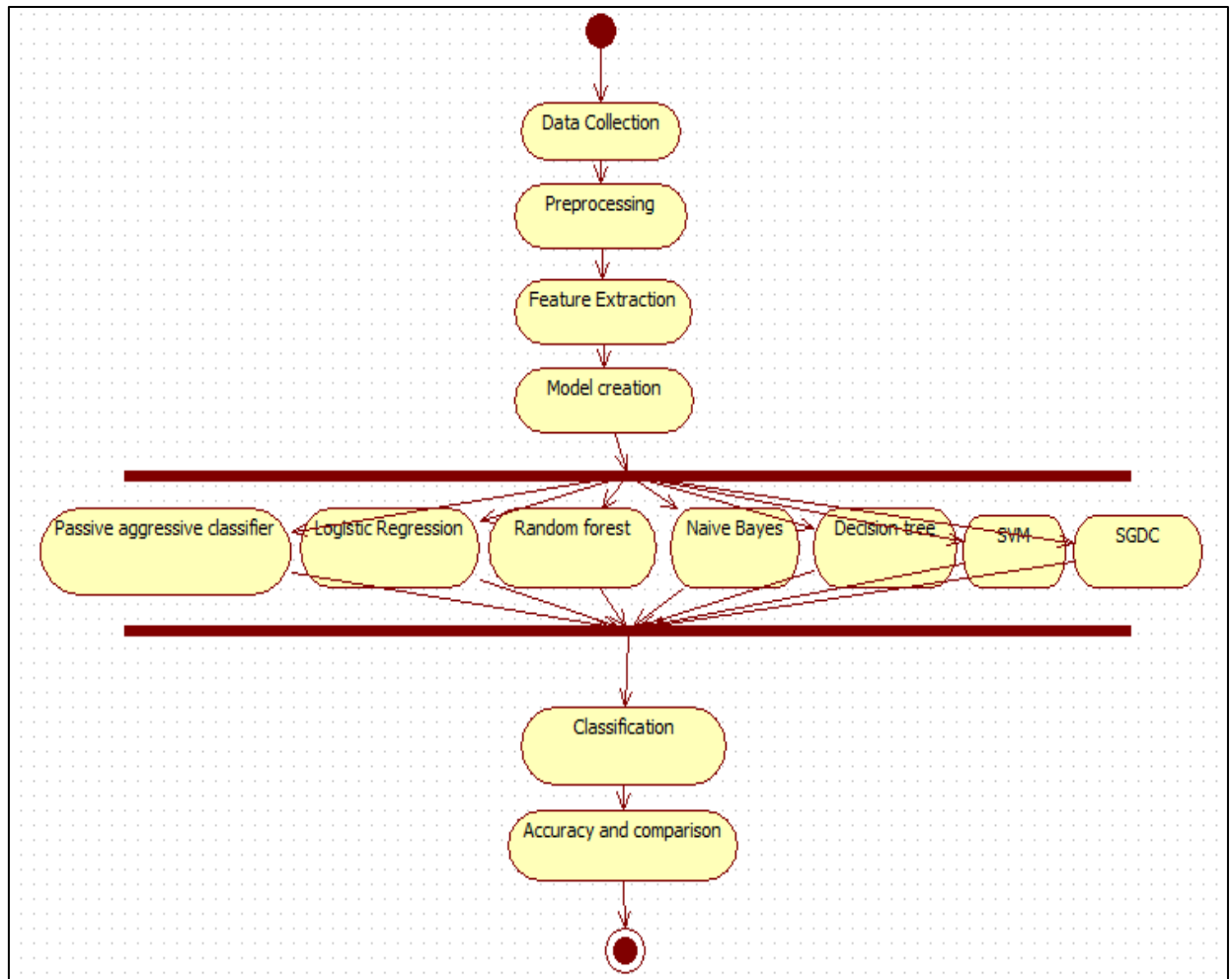


Fig: 3.6 Activity Diagram

### 3.13 METHODOLOGIES

#### 3.13.1 Data collection:

The dataset was collected in kaggle website (**news.csv**). This dataset has a shape of 7796×4. The first column identifies the news, the second and third are the title and text, and the fourth column has labels denoting whether the news is REAL or FAKE.

#### 3.13.2 Data Preprocessing:

This is the initial stage which convert our raw data into cleaned data. When collecting data from different source which is not in proper format it is not feasible for better prediction. Now, let's read the data into a DataFrame, and get the shape of the data and the first 5 records.

### **3.13.3 Feature Extraction:**

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. get the labels from the DataFrame. Split the dataset into training and testing sets. initialize a TfidfVectorizer with stop words from the English language and a maximum document frequency of 0.7 (terms with a higher document frequency will be discarded). Stop words are the most common words in a language that are to be filtered out before processing the natural language data. And a TfidfVectorizer turns a collection of raw documents into a matrix of TF-IDF features. Now, fit and transform the vectorizer on the train set, and transform the vectorizer on the test set.

### **3.13.4 Model creation:**

Split the dataset into training and testing sets. we present an analysis of fake news detection using various classifiers like Passive aggressive Classifier, Naïve Bayes, Random forest, Support Vector Machine (SVM), stochastic gradient descent, Logistic regression and decision tree classifier.

### **3.13.5 Classification:**

We Classifies how much real and fake present in dataset. Graphical representation of Real and Fake news. We got an accuracy with this all model. Finally, print out a confusion matrix to gain insight into the number of false and true negatives and positives. we get the accuracy and compared result.

#### **3.13.5.1 Algorithms:**

##### **1. Naïve Bayes Classifier:**

Naïve Bayes classifiers are a set of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms that share a common standard principle, i.e. every pair of features being classified is independent of each other. Naïve Bayes classifiers assume strong, or Naïve, independence between attributes of knowledge points. Naïve Bayes assumes the independence of features which is not true for news articles as words usually come together.

Equation:

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

Using Bayes theorem, we will find the probability of an event as long as B has occurred. Here, B is that evidence and A is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of single particular feature does not affect the other. Hence it is called Naïve Bayes.

## **2. Logistic Regression:**

Logistic regression is a type of statistical analysis that is used to predict the outcome of a dependent variable based on prior observations. It is mainly used for binary classification. A logistic regression algorithm looks at the connection between a dependent variable and one or more dependent variables. Logistic regression needs a large data set for higher accuracy while Naïve Bayes can work with small datasets too. 3.3 Decision Trees Decision Tree solves the problem of machine learning by transforming the information into tree representation. Each internal node of the tree representation denotes an attribute and every leaf node denotes as class label. Decision tree algorithm are often used for solving both regression and classification problems. A decision tree would over fit when there are a large number of sparse features, and therefore perform poorly on the testing data.

## **3. Support Vector Machines**

SVM is also a supervised machine learning algorithm that is oftently used for both classification and regression purposes. The support vector machine performs well when there is an extremely high number of features in a text classification problem but generally requires a lot of tuning and is memory intensive. Once we have labelled training data (supervised learning), the algorithm generates the best possible hyperplane which categorizes new data automatically.

## **4. Decision Tree**

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes

represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees

## **5. Random Forest**

- Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we've collection of decision trees (so known as "Forest"). To classify a new object based on attributes, each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).
- Each tree is planted & grown as follows:
- If the number of cases in the training set is  $N$ , then sample of  $N$  cases is taken at random but with replacement. This sample will be the training set for growing the tree.
- If there are  $M$  input variables, a number  $m \ll M$  is specified such that at each node,  $m$  variables are selected at random out of the  $M$  and the best split on these  $m$  is used to split the node. The value of  $m$  is held constant during the forest growing.

Each tree is grown to the largest extent possible. There is no pruning.

## 6. Passive-Aggressive algorithms

This is a high-level overview of the algorithm explaining how it works and when to use it. It does not go deep into the mathematics of how it works. Passive-Aggressive algorithms are generally used for large-scale learning. It is one of the few 'online-learning algorithms'. In online machine learning algorithms, the input data comes in sequential order and the machine learning model is updated step-by-step, as opposed to batch learning, where the entire training dataset is used at once. This is very useful in situations where there is a huge amount of data and it is computationally infeasible to train the entire dataset because of the sheer size of the data. We can simply say that an online-learning algorithm will get a training example, update the classifier, and then throw away the example.

- A very good example of this would be to detect fake news on a social media website like Twitter, where new data is being added every second. To dynamically read data from Twitter continuously, the data would be huge, and using an online-learning algorithm would be ideal.
- Passive-Aggressive algorithms are somewhat similar to a Perceptron model, in the sense that they do not require a learning rate. However, they do include a regularization parameter.

## 7. Stochastic Gradient descent

- Gradient descent is an iterative algorithm, that starts from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function.”
- This algorithm is useful in cases where the optimal points cannot be found by equating the slope of the function to 0. In the case of linear regression, you can mentally map the sum of squared residuals as the function “y” and the weight vector as “x” in the parabola above.
- This is the crux of the algorithm. The general idea is to start with a random point (in our parabola example start with a random “x”) and find a way to update this point with each iteration such that we descend the slope.
- The steps of the algorithm are Find the slope of the objective function with respect to each parameter/feature. In other words, compute the gradient of the function.

- Pick a random initial value for the parameters. (To clarify, in the parabola example, differentiate “y” with respect to “x”. If we had more features like x1, x2 etc., we take the partial derivative of “y” with respect to each of the features.)
- Update the gradient function by plugging in the parameter values.
- Calculate the step sizes for each feature as :  $\text{step size} = \text{gradient} * \text{learning rate}$ .
- Calculate the new parameters as :  $\text{new params} = \text{old params} - \text{step size}$
- Repeat steps 3 to 5 until gradient is almost 0.

### 3.14 EVALUATION METRICS

To evaluate the performance of algorithms for fake news detection problem, various evaluation metrics are used. Here, we review the most widely used metrics for fake news detection. Most of the existing models consider the fake news problem as a classification problem that predicts whether a news is fake or not: □

- True Positive (TP): when predicted fake news article or report are actually annotated as fake news.
- True Negative (TN): when predicted true news article or report are actually annotated as true news. □
- False Negative (FN): when predicted true news article or report are actually annotated as fake news.
- False Positive (FP): when predicted fake news article or report are actually annotated as true news.

### 3.15 TESTING

#### 3.15.1 Software Testing

##### General

In a generalized way, we can say that the system testing is a type of testing in which the main aim is to make sure that system performs efficiently and seamlessly. The process of testing is applied to a program with the main aim to discover an unprecedented error, an error which otherwise could have damaged the future of the software. Test cases which brings up a high possibility of discovering an error is considered successful. This successful test helps to answer the still unknown errors.

### **3.15.2 TEST CASE**

Testing, as already explained earlier, is the process of discovering all possible weak-points in the finalized software product. Testing helps to counter the working of sub-assemblies, components, assembly and the complete result. The software is taken through different exercises with the main aim of making sure that software meets the business requirement and user-expectations and doesn't fail abruptly. Several types of tests are used today. Each test type addresses a specific testing requirement.

### **3.15.3 Testing Techniques**

A test plan is a document which describes approach, its scope, its resources and the schedule of aimed testing exercises. It helps to identify almost other test item, the features which are to be tested, its tasks, how will everyone do each task, how much the tester is independent, the environment in which the test is taking place, its technique of design plus the both the end criteria which is used, also rational of choice of theirs, and whatever kind of risk which requires emergency planning. It can be also referred to as the record of the process of test planning. Test plans are usually prepared with signification input from test engineers.

### **(I) UNIT TESTING**

In unit testing, the design of the test cases is involved that helps in the validation of the internal program logic. The validation of all the decision branches and internal code takes place. After the individual unit is completed it takes place. Plus it is taken into account after the individual unit is completed before integration. The unit test thus performs the basic level test at its component stage and test the particular business process, system configurations etc. The unit test ensures that the particular unique path of the process gets performed precisely to the documented specifications and contains clearly defined inputs with the results which are expected.

### **(II) INTEGRATION TESTING**

These tests are designed to test the integrated software items to determine whether if they really execute as a single program or application. The testing is

event driven and thus is concerned with the basic outcome of field. The Integration tests demonstrate that the components were individually satisfaction, as already represented by successful unit testing, the components are apt and fine. This type of testing is specially aimed to expose the issues that come-up by the components combination.

### **(III) FUNCTIONAL TESTING**

The functional tests help in providing the systematic representation that functions tested are available and specified by technical requirement, documentation of the system and the user manual.

### **(IV) SYSTEM TESTING**

System testing, as the name suggests, is the type of testing in which ensure that the software system meet the business requirements and aim. Testing of the configuration is taken place here to ensure predictable result and thus analysis of it. System testing is relied on the description of process and its flow, stressing on pre driven process and the points of integration.

### **(V) WHITE BOX TESTING**

The white box testing is the type of testing in which the internal components of the system software is open and can be processed by the tester. It is therefore a complex type of testing process. All the data structure, components etc. are tested by the tester himself to find out a possible bug or error. It is used in situation in which the black box is incapable of finding out a bug. It is a complex type of testing which takes more time to get applied.

### **(VI) BLACK BOX TESTING**

The black box testing is the type of testing in which the internal components of the software is hidden and only the input and output of the system is the key for the tester to find out a bug. It is therefore a simple type of testing. A programmer with basic knowledge can also process this type of testing. It is less time consuming as compared to the white box testing. It is very successful for software which are less complex are straight-forward in nature. It is also less costly than white box testing.

## **(VII) ACCEPTANCE TESTING**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

## CHAPTER 4

### RESULT AND DISCUSSION

#### 4.1 RESULTS

In this project finally comparing dataset with combination of real and fake news with various algorithm using vectorizer TF\_IDF vectorizer and Count Vectorizer, we can say that Stochastic Gradient Descent is the classification algorithm that give best accuracy and performance. Here we first implement this with PassiveAggressive Classifier and other. Then we found usage of Stochastic Gradient descent will gives uses more efficiency. One of the main advantage of SGDClassifier is a best efficiency.

Here comparison of passiveAggressive Classifier and Logistic Regression with accuracy of 92.74% and 91.55%

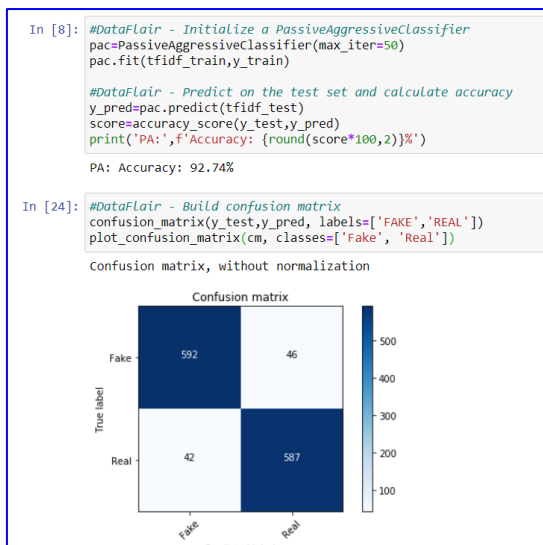


Fig 4.1 Output for PA

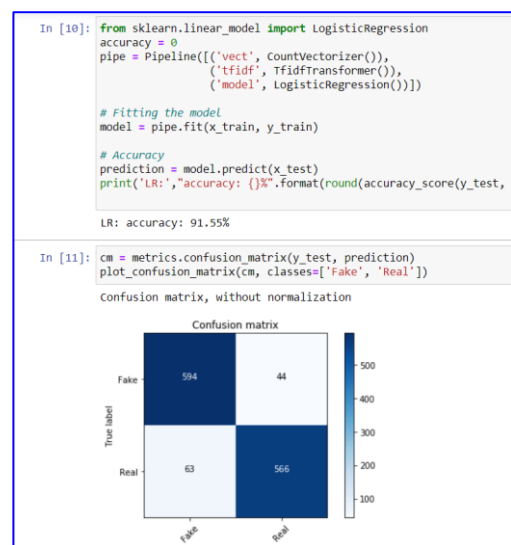


Fig 4.2 Output for LR

Then DecisionTree Classifier and RandomForestClassifier as been used which gives us less accuracy compared to PA and LR then, here we can say that DT gives us the least accuracy comparing all in our dataset.

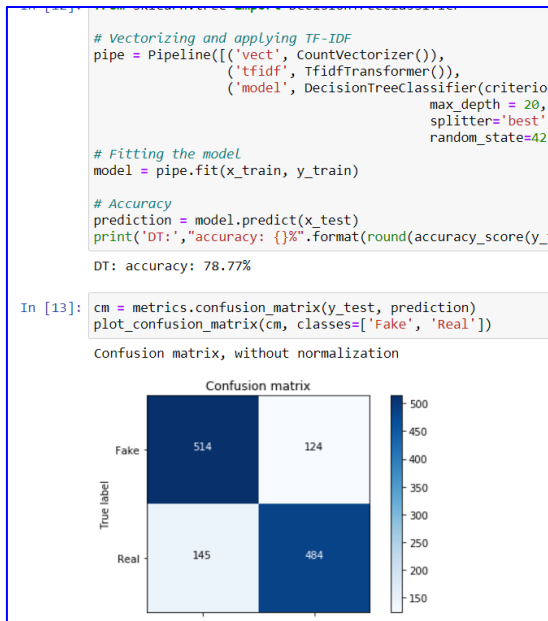


Fig 4.3 Output for DT

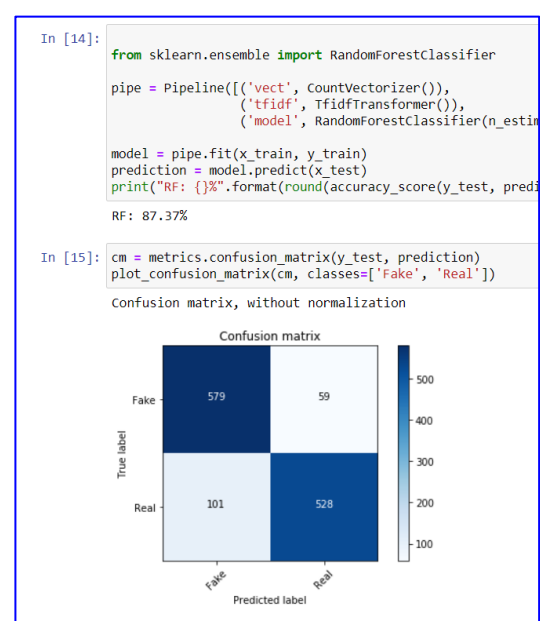


Fig 4.4: Output for RFC

Then Multinomial Naïve Bayes Classifier is used that about higher accuracy is some dataset but here is not the least and the percent of finding accuracy is much higher to DT. The SVM is used for comparison which is nearly equal to PA

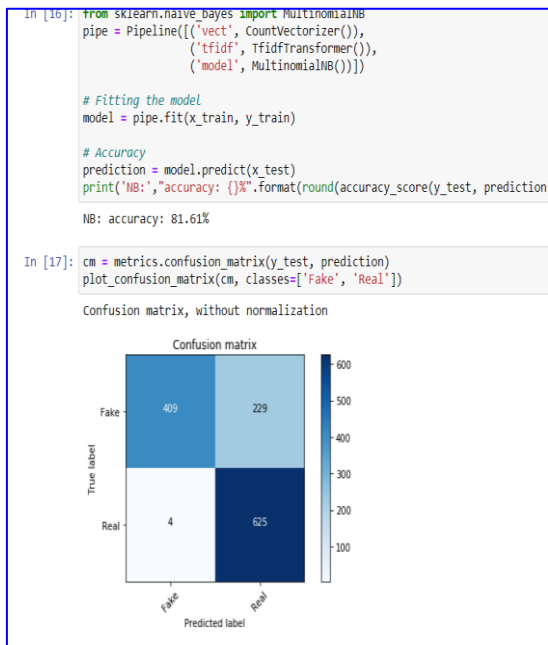


Fig 4.5 Output for NB

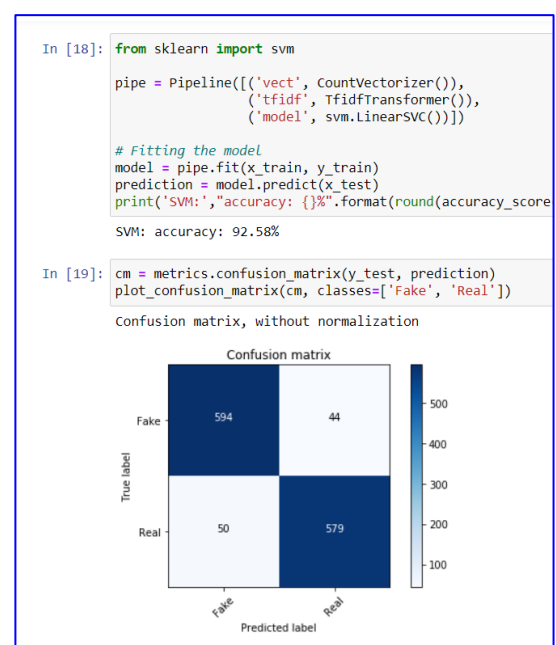


Fig 4.6 Output for SVM

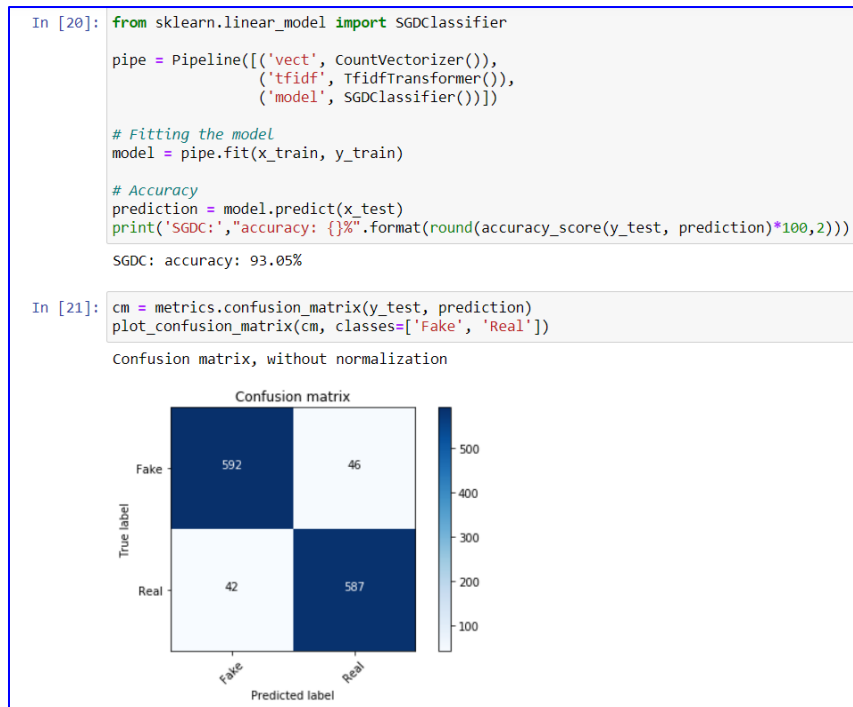


Fig 4.7 Output for SGDC

This is the new Classifier we used then obtain and gave us the highest accuracy SGDC. The overall comparison table and graph will help us in understanding the comparison between the algorithms more clearly.

| Classification Algorithms | DecisionTree Classifier | NaiveBayes Classifier | RandomForest Classifier | LogisticRegression | Support vector mahine | PassiveAggressive Classifier | Stochastic Gradient Descent Classifier |
|---------------------------|-------------------------|-----------------------|-------------------------|--------------------|-----------------------|------------------------------|----------------------------------------|
| Accuracy                  | 78.77                   | 81.66                 | 87.69                   | 91.55              | 91                    | 92.58                        | 93.05                                  |

Table 4.1 Overall Accuracy

4.2 COMPARISON GRAPH:

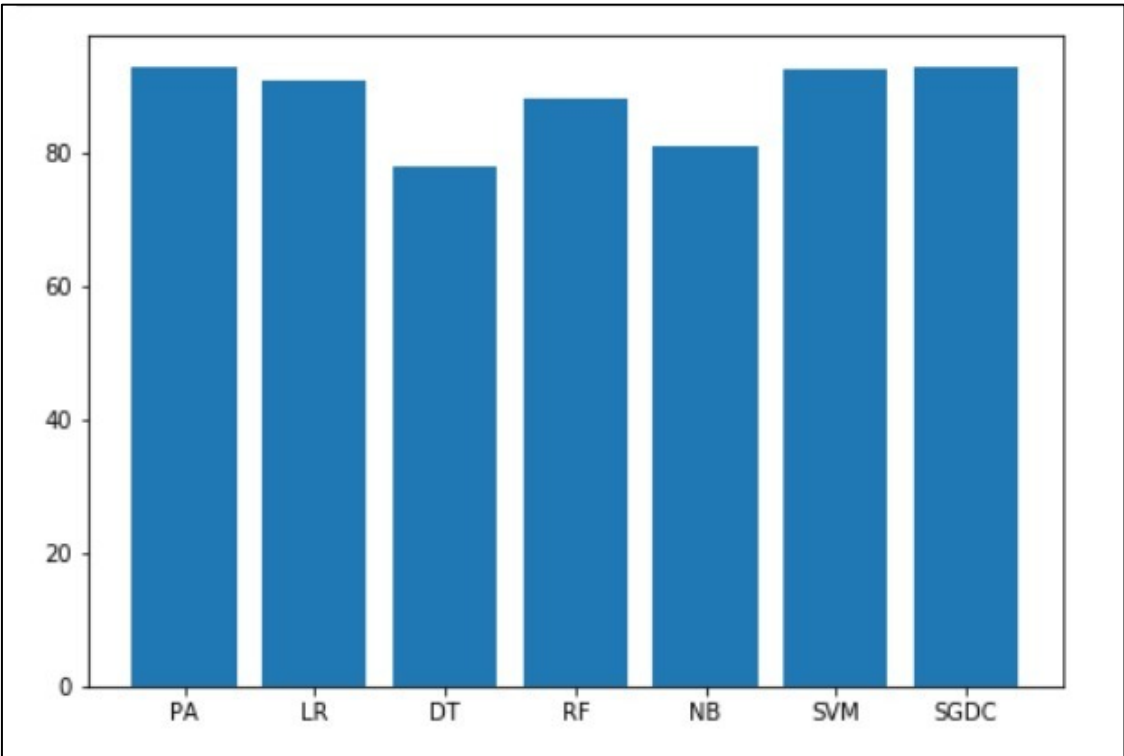


Fig 4.8 Overall Accuracy Graph 1

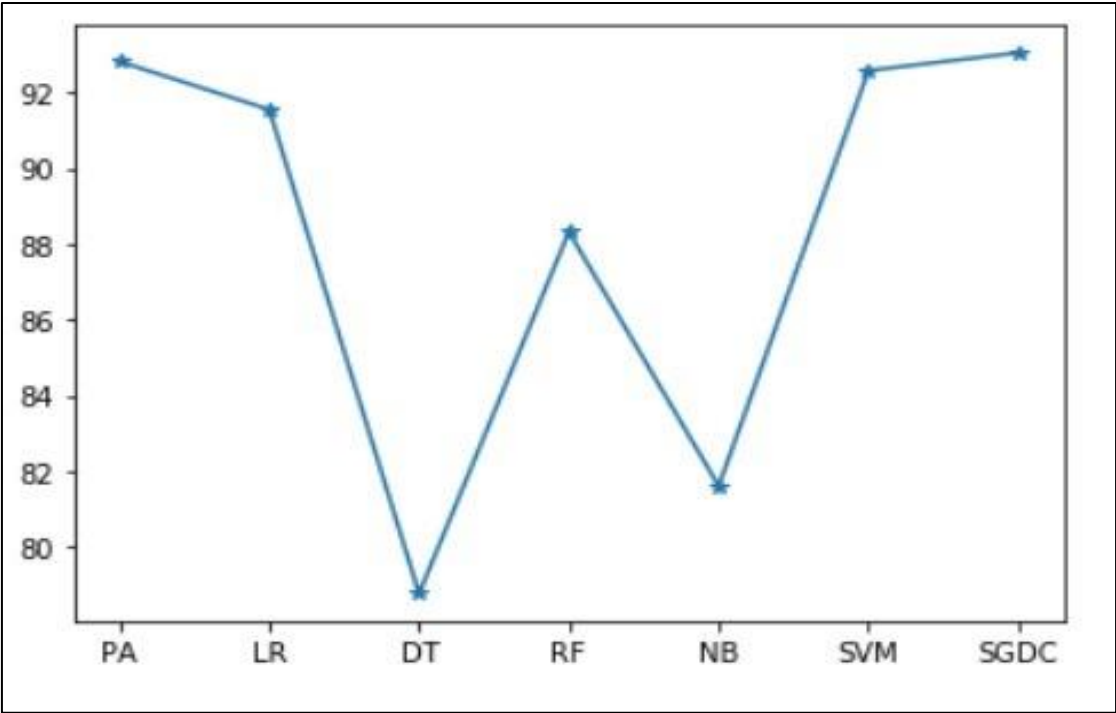


Fig 4.9 Overall Accuracy Graph 2

## **CHAPTER 5**

### **CONCLUSION AND ENHANCEMENT**

#### **5.1 CONCLUSION**

Scrutinizing fake news on social media using machine learning techniques compares different classification methods, for predicting accuracy among the proposed classifiers. Based on which the content provided by the dataset, classification using various machine learning algorithms. In this project, we aim to predict accuracy by comparing various classification algorithms such as Decision Tree, Naïve Bayes, Random Forest, Logistic Regression, Support Vector Machine, Passive Aggressive Classifier, Stochastic Gradient Descent Classifier.

While comparing all, Decision Tree Classifier gives the less accuracy than all while passive Aggressive Classifier, Support Vector Machine, Stochastic Gradient Descent Classifier are nearly same, but it has been proved that Stochastic Gradient Descent Classifier works best than all other algorithms which has been build on TF-IDF feature vectorizer. We are initializing the data, then by using TF-IDF Vectorizer it will split the training and testing data. The model is trained by different classification algorithms then testing data is trained and validated so that resulting accuracy with its model can be clearly defined. Finally, the outcome of this project shows that the Stochastic Gradient Descent Classifier has better accuracy and performance than other classification algorithms like PassiveAggressive, LogisticRegression, etc on the dataset. It can be perceived from a classification report. So that we can conclude by saying accuracy from fake news has been detected correctly.

#### **5.2 FUTURE WORKS**

In future we can work on hybrid model using neural network and similarity for better performance. Our approach can be further extended to detection the fake news.

## REFERENCES

- [1] "CSI: A Hybrid Deep Model for Fake News Detection," CIKM17, Singapore, Singapore. Author: Natali Ruchansky, Sungyong Seo and Yan Liu Year: 2017
- [2] "Automatically Identifying Fake News in Popular Twitter Threads," 2017 IEEE International Conference on Smart Cloud. Author: Cody Buntain and Jennifer Golbeck Year: 2017
- [3] "Spotting Fake News: A Social Argumentation Framework for Scrutinizing Alternative Facts," 2017 IEEE 24th International Conference on Web Services. Author: Ricky J. Sethi Year: 2017
- [4] N. Smitha; R. Bharath "Performance Comparison of Machine Learning Classifiers for Fake News Detection" 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA) Year: 2020
- [5] "Automatic Online Fake News Detection Combining Content and Social Signals," PROCEEDING OF THE 22ND CONFERENCE OF FRUCT ASSOCIATION. Author: Marco L. Della Vedova, Eugenio Tacchini, Ballarin Year: 2018
- [6] "CIMTDetect: A Community Infused Matrix-Tensor Coupled Factorization Based Method for Fake News Detection," 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). Author: Shashan Gupta, Raghuveer Thirukovalluru Year: 2018
- [7] "Identifying Tweets With Fake News," 2018 IEEE International Conference on Information Reuse and Integration for Data Science. Author: Saranya Krishnan and Min Chen Year: 2018
- [8] "Understanding User Profiles on Social Media for Fake News Detection," 2018 IEEE Conference on Multimedia Information Processing and Retrieval. Author: Kai Shu, Suhag Wang and Huan Liu Year: 2018
- [9] "Hybrid Machine-Crowd Approach for Fake News Detection," 2018 IEEE 4th International Conference on Collaboration and Internet Computing. Author: Shaban Shabani and Maria Sokhn Year: 2018
- [10] "Fake News Detection Enhancement with Data Imputation," 2018 IEEE 16th Int. Conf. on Dependable, Autonomic Secure Comp., 16th Int. Conf. on Pervasive Intelligence Comp., 4th Int. Conf. on Big Data Intelligence Comp., and 3rd Cyber

Sci. Tech. Cong. Author:Chandra Mouli Madhav Kotteti, Xishuang Dong, Na Li and Lijun Qian Year:2018

[11] Akshay Jain, Amey Kasbe, “ Fake News Detection”, 2018 IEEE International conference for student, Electrical and Electronics and computer science

[12] Ehesas Mia Mahir, Abdullah-All-Tanvir,” Detecting Fake News using Machine Learning and Deep Learning Algorithm”, 2019 7th International Conference on Smart Computing.

[13] M. Granik and V. Mesyura, “Fake news detection using Naïve Bayes classifier,” 2017 IEEE 1st Ukr. Conf. Electr. Comput. Eng. UKRCON 2017 - Proc.,

[14] “Weakly Supervised Learning for Fake News Detection on Twitter,” 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). Author:Stefan Helmstetter and Heiko Paulheim Year:2018

[15] Rahul R Mandical;N Mamatha;N Shivakumar;R Monica;A N Krishna “Identification of Fake News Using Machine Learning” 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) Year: 2020

[16] M. Aldwairi and A. Alwahedi, “Detecting Fake News in Social Media Networks”, The 9th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN) 2018.

[17] <https://data-flair.training/blogs/advanced-python-project-detecting-fake-news>

[18] <https://www.kaggle.com/davidmercadofaustino/fake-news-detection>

## APPENDICES

### A. SOURCE CODE

```
import numpy as np
import pandas as pd
import itertools
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.pipeline import Pipeline

#Read the data
df=pd.read_csv(r'C:\Users\DELL\Desktop\DFN_CODE\main_jenila\news.csv')

#Get shape and head
df.shape
df.head(5)

#DataFlair - Get the labels
labels=df.label
real = 0
fake = 0
for i in labels:
    if i == "REAL":
        real = real+1
    else:
        fake = fake+1
print('REAL:',real)
print("FAKE:",fake)
```

```
# FAKE_NUM = labels.count()
# PRINT(FAKE_NUM)
```

```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
status = ['REAL','FAKE']
value = [real,fake]
ax.bar(status,value)
```

```
plt.show()
```

```
#DataFlair - Split the dataset
```

```
x_train,x_test,y_train,y_test=train_test_split(df['text'], labels, test_size=0.2,
random_state=7)
```

```
#DataFlair - Initialize a TfidfVectorizer
```

```
tfidf_vectorizer=TfidfVectorizer(stop_words='english', max_df=0.7)
```

```
#DataFlair - Fit and transform train set, transform test set
```

```
tfidf_train=tfidf_vectorizer.fit_transform(x_train)
```

```
tfidf_test=tfidf_vectorizer.transform(x_test)
```

```
from sklearn import metrics
```

```
import itertools
```

```
def plot_confusion_matrix(cm, classes,
                           normalize=False,
                           title='Confusion matrix',
                           cmap=plt.cm.Blues):
```

```
plt.imshow(cm, interpolation='nearest', cmap=cmap)
```

```

plt.title(title)
plt.colorbar()
tick_marks = np.arange(len(classes))
plt.xticks(tick_marks, classes, rotation=45)
plt.yticks(tick_marks, classes)

if normalize:
    cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
    print("Normalized confusion matrix")
else:
    print('Confusion matrix, without normalization')

    thresh = cm.max() / 2.
for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
    plt.text(j, i, cm[i, j],
             horizontalalignment="center",
             color="white" if cm[i, j] > thresh else "black")

plt.tight_layout()
plt.ylabel('True label')
plt.xlabel('Predicted label')

#DataFlair - Initialize a PassiveAggressiveClassifier
pac=PassiveAggressiveClassifier(max_iter=50)
pac.fit(tfidf_train,y_train)

#DataFlair - Predict on the test set and calculate accuracy
y_pred=pac.predict(tfidf_test)
score=accuracy_score(y_test,y_pred)
print('PA:',f'Accuracy: {round(score*100,2)}%')

#DataFlair - Build confusion matrix
confusion_matrix(y_test,y_pred, labels=['FAKE','REAL'])
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

```

```

from sklearn.linear_model import LogisticRegression
accuracy = 0
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', LogisticRegression())])

# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('LR:', "accuracy:{}".format(round(accuracy_score(y_test,
prediction)*100,2)))

cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

from sklearn.tree import DecisionTreeClassifier

# Vectorizing and applying TF-IDF
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', DecisionTreeClassifier(criterion= 'entropy',
                                                    max_depth = 20,
                                                    splitter='best',
                                                    random_state=42))])

# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('DT:', "accuracy:{}".format(round(accuracy_score(y_test,
prediction)*100,2)))

```

```

cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

from sklearn.ensemble import RandomForestClassifier

pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', RandomForestClassifier(n_estimators=50,
criterion="entropy"))])

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

cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

from sklearn.Naïve_bayes import MultinomialNB
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', MultinomialNB())])

# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('NB:', "accuracy: {}".format(round(accuracy_score(y_test,
prediction)*100,2)))

cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

```

```

from sklearn import SVN

pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', SVN.LinearSVC())])

# Fitting the model
model = pipe.fit(x_train, y_train)
prediction = model.predict(x_test)
print('SVM:', "accuracy:          {}".format(round(accuracy_score(y_test,
prediction)*100,2)))

cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

from sklearn.linear_model import SGDClassifier

pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', SGDClassifier())])

# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('SGDC:', "accuracy:          {}".format(round(accuracy_score(y_test,
prediction)*100,2)))

cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

fig = plt.figure()

```

```

ax.set_title('a')
ax = fig.add_axes([1,1,1,1])
status = ['PA','LR','DT','RF','NB','SVM','SGDC']
value = [92.82,91.55,78.77,88.32,81.61,92.58,93.05]
ax.bar(status,value)

```

```

plt.show()

```

```

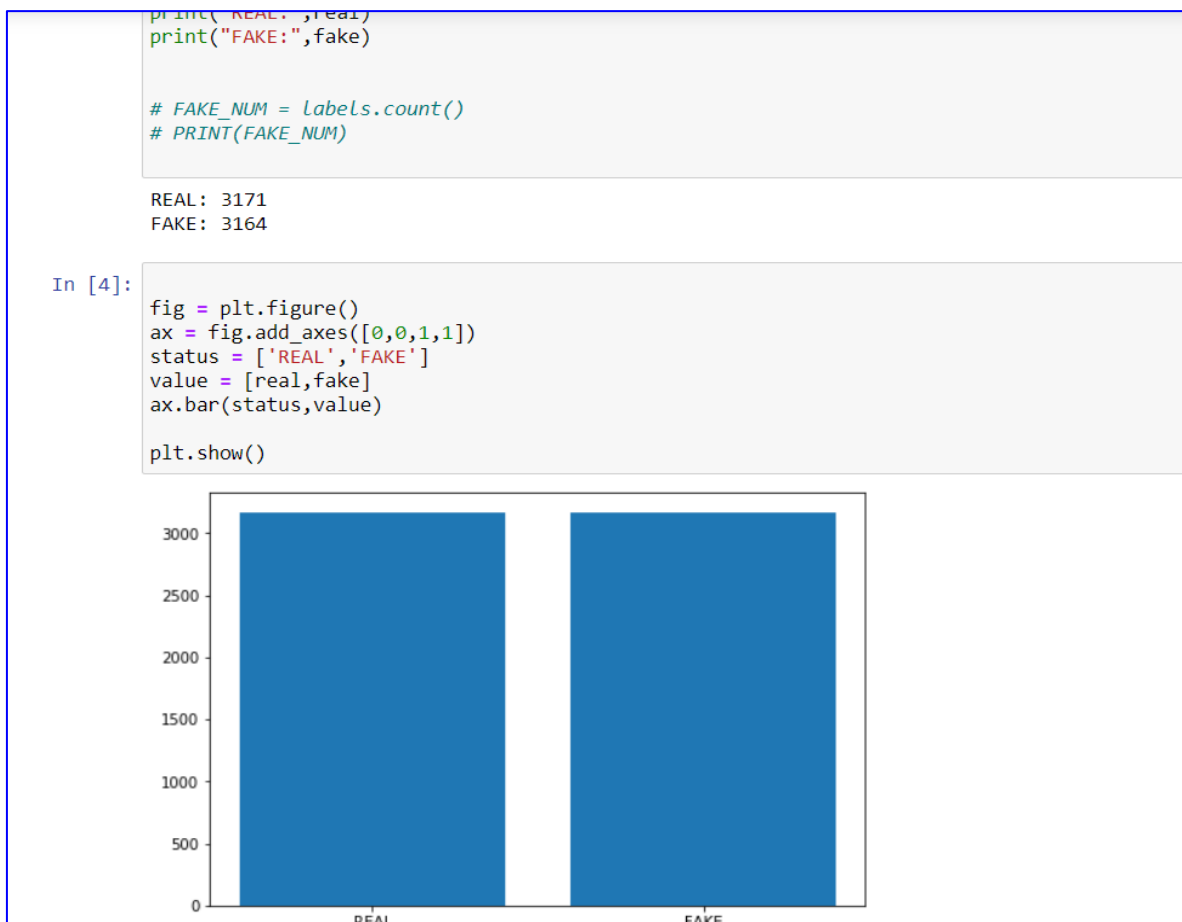
a= plt.figure()
# adding axes
axes= a.add_axes([0.1,0.1,0.8,0.8])
x= np.arange(0,11)
axes.plot(status,value, marker='*')
status = ['PA','LR','DT','RF','NB','SVM','SGDC']
value = [92.82,91.55,78.77,88.32,81.61,92.58,93.05]
plt.show()

```

## B. SCREENSHOTS



Fig



Fig

```
In [5]: #DataFlair - Split the dataset
x_train,x_test,y_train,y_test=train_test_split(df['text'], labels, test_size=0.2, random_state=7)
```

```
In [6]: #DataFlair - Initialize a TfidfVectorizer
tfidf_vectorizer=TfidfVectorizer(stop_words='english', max_df=0.7)

#DataFlair - Fit and transform train set, transform test set
tfidf_train=tfidf_vectorizer.fit_transform(x_train)
tfidf_test=tfidf_vectorizer.transform(x_test)
```

```
In [7]: from sklearn import metrics
import itertools

def plot_confusion_matrix(cm, classes,
                           normalize=False,
                           title='Confusion matrix',
                           cmap=plt.cm.Blues):

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
```

Fig

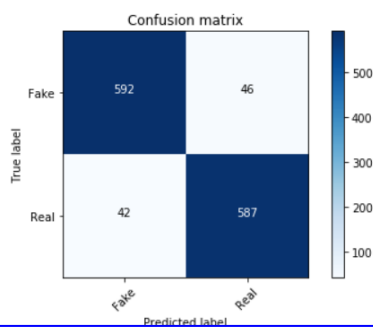
```
In [8]: #DataFlair - Initialize a PassiveAggressiveClassifier
pac=PassiveAggressiveClassifier(max_iter=50)
pac.fit(tfidf_train,y_train)

#DataFlair - Predict on the test set and calculate accuracy
y_pred=pac.predict(tfidf_test)
score=accuracy_score(y_test,y_pred)
print('PA:',f'Accuracy: {round(score*100,2)}%')
```

PA: Accuracy: 92.74%

```
In [24]: #DataFlair - Build confusion matrix
confusion_matrix(y_test,y_pred, labels=['FAKE', 'REAL'])
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

Confusion matrix, without normalization



```
In [10]: from sklearn.linear_model import LogisticRegression
accuracy = 0
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', LogisticRegression())])

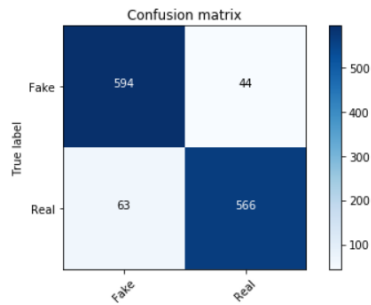
# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('LR:', "accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
```

LR: accuracy: 91.55%

```
In [11]: cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

Confusion matrix, without normalization



```
In [12]: from sklearn.tree import DecisionTreeClassifier

# Vectorizing and applying TF-IDF
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', DecisionTreeClassifier(criterion='entropy',
                                                  max_depth=20,
                                                  splitter='best',
                                                  random_state=42))])

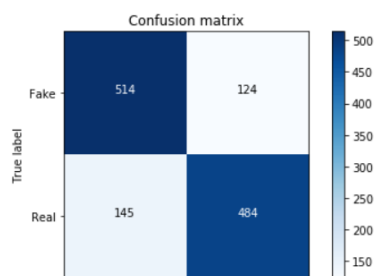
# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('DT:', "accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))
```

DT: accuracy: 78.77%

```
In [13]: cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])
```

Confusion matrix, without normalization



```
In [14]: from sklearn.ensemble import RandomForestClassifier

pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', RandomForestClassifier(n_estimators=50, criterion="entropy"))])

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

RF: 87.37%
```

```
In [15]: cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

Confusion matrix, without normalization
```



```
In [16]: from sklearn.naive_bayes import MultinomialNB

pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', MultinomialNB())])

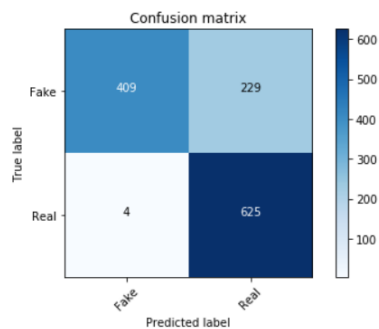
# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('NB:', "accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))

NB: accuracy: 81.61%
```

```
In [17]: cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

Confusion matrix, without normalization
```



```
In [18]: from sklearn import svm

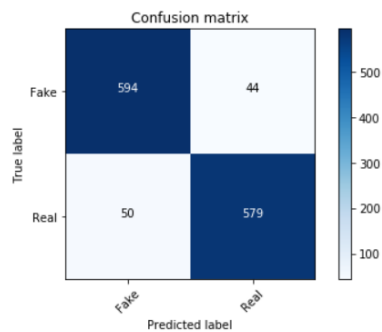
pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', svm.LinearSVC())])

# Fitting the model
model = pipe.fit(x_train, y_train)
prediction = model.predict(x_test)
print('SVM:', "accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))

SVM: accuracy: 92.58%
```

```
In [19]: cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

Confusion matrix, without normalization
```



```
In [20]: from sklearn.linear_model import SGDClassifier

pipe = Pipeline([('vect', CountVectorizer()),
                  ('tfidf', TfidfTransformer()),
                  ('model', SGDClassifier())])

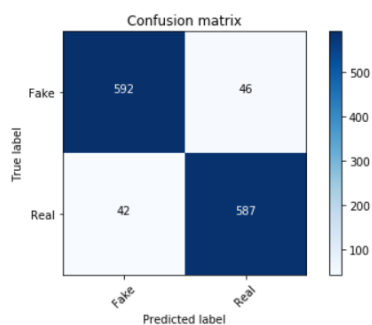
# Fitting the model
model = pipe.fit(x_train, y_train)

# Accuracy
prediction = model.predict(x_test)
print('SGDC:', "accuracy: {}".format(round(accuracy_score(y_test, prediction)*100,2)))

SGDC: accuracy: 93.05%
```

```
In [21]: cm = metrics.confusion_matrix(y_test, prediction)
plot_confusion_matrix(cm, classes=['Fake', 'Real'])

Confusion matrix, without normalization
```



## C. PLAGARISM REPORT

### Scrutinizing of Fake News using Machine Learning Techniques

**8** Dr.S.Gowri  
Department of Information  
Technology  
Sathyabama Institute of  
Science and Technology,  
Chennai, India.  
[gowriamirtha2003@gmail.com](mailto:gowriamirtha2003@gmail.com)

**8** Jenila J  
Department of Information  
Technology  
Sathyabama Institute of  
Science and Technology,  
Chennai, India  
[justinjenila@gmail.com](mailto:justinjenila@gmail.com)

**8** athula Sowmya Reddy  
Department of Information  
Technology  
Sathyabama Institute of  
Science and Technology,  
Chennai, India  
[sowmyareddyathula@gmail.com](mailto:sowmyareddyathula@gmail.com)

M.Antony Sheela  
System Administrator  
GPR Nata Training Center  
Nagercoil.  
[antonysheelam@gmail.com](mailto:antonysheelam@gmail.com)

**Abstract**—Fake news is false or misleading information, that has the aim of damaging the personality of people just for making money. Fake news spreads and reduces the impact of real news. Fake news comprises of news that is not been properly researched and proved information spreads false alarm via varying form in social media. People not only make verdict out information which are extracted on the basis of social media even they also assemble their own decisions out of it. In order to confront that problem a computational model for likelihood machine learning technique that detects fake news. We build TF-IDF vectorizer on our dataset that uses classification algorithm like SGDClassifier (Stochastic Gradient Descent Classifier) for better efficiency, PassiveAggressive Classifier etc. The TfidfVectorizer (Term Frequency Inverse Document Frequency) gives classification accuracy of the model and converts a raw document into a confusion matrix of TF-IDF Vectorizer. After all, the accuracy score let us know how far the project is implemented. The proposed model defines correctness of results for maximum accuracy.

**Keywords**—Fake news, Social media, TF-IDF Vectorizer, PassiveAggressive Classifier, Machine learning.

social media. Detecting fake news is not a easy task, using various algorithms we can predict the accuracy of the fake news. In this paper we perform analysis among various classification algorithms such as Decision Tree, Naive Bayes, Random Forest, Logistic Regression, Support Vector Machine, Passive Aggressive, Stochastic Gradient Descent Classifier.

The main objectives of fake news detection is to determine high accuracy of news whether it is fake or not. Using machine learning model we can accurately determine the fake news.[4]

#### II. LITERATURE REVIEW

Natali Ruchansky et al [1] This Model is spurred by three quality of fake news as for example content of the article, the client reaction it get, and the source client advancing it. These three quality are made up of three varying modules as for instance Capture, Score and Integration. Capture counts on reaction and content. To remove transient portrayal of articles using Recurrent Neural Network (RNN). Score module studies the source quality about the conduct of

### Scrutinizing of Fake News Using Machine Learning Techniques

#### ORIGINALITY REPORT

|                  |                  |              |                |
|------------------|------------------|--------------|----------------|
| <b>15%</b>       | <b>10%</b>       | <b>3%</b>    | <b>%</b>       |
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

#### PRIMARY SOURCES

|          |                                                                                                                                                                                                                                      |           |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|
| <b>1</b> | <b>www.ijeat.org</b><br>Internet Source                                                                                                                                                                                              | <b>4%</b> |
| <b>2</b> | <b>Ashokkumar Thakur, Sujit Shinde, Tejas Patil, Brijesh Gaud, Vanita Babanne. "MYTHYA: Fake News Detector, Real Time News Extractor and Classifier", 2020 4th International Conference on Trends in Electronics and Informatics</b> | <b>4%</b> |