A PROJECT REPORT ON

**Fake Instagram Profile Identification and Classification using Machine Learning**

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE OF

## BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

**SUBMITTED BY**

**STUDENT NAME SEAT NO**

### Apurv Kamalakar Badave B190364217

**Aniruddha Shivaji Lalge B190364389**

### Nikhil Lalasaheb Elajale B190364283

**Pankaj Dayaram Godara B190364304**



**DEPARTMENT OF COMPUTER ENGINEERING**

**STES’S SMT. KASHIBAINAVALECOLLEGE OF ENGINEERING VADGAON BK, OFF SINHGAD ROAD, PUNE 411041**

SAVITRIBAI PHULE PUNE UNIVERSITY

**2023-2024**



## CERTIFICATE

This is to certify that the project report entitles

## “Fake Instagram Profile Identification and Classification using Machine Learning” Submitted by

### Apurv Kamalakar Badave Seat No: B190364217

**Aniruddha Shivaji Lalge Seat No:** **B190364389**

### Nikhil Lalasaheb Elajale Seat No: B190364283

**Pankaj Dayaram Godara Seat No: B190364304**

is a bonafide work of this institute and the work has been carried out by them under the supervision of **Prof. Priyanka Kinage** and it is approved for the partial fulfilment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering)

**(Prof. Priyanka Kinage) (Prof. R. H. Borhade)**

Internal Guide Head,

Department of Computer Engineering Department of Computer Engineering

**(Dr. A. V. Deshpande)**

Principal,

Smt. Kashibai Navale College of Engineering Pune – 41

# ACKNOWLEDGEMENT

It gives us great pleasure in presenting the preliminary project report on **‘Fake Instagram Profile Identification and Classification using Machine Learning’**. We would like to take this opportunity to thank our internal guide **Prof. Priyanka Kinage** for giving us all the help and guidance we needed. We are really grateful to her for their kind support. Their valuable suggestions were very helpful.

We are also grateful to **Prof. R. H. Borhade**, Head of Computer Engineering Department, SKNCOE for his indispensable support, suggestions.

We would like to convey our gratitude to **Dr. A. V. Deshpande (Principal)** all the teaching and non-teaching staff members of the Computer Engineering Department who gave us the freedom to explore and guided us the right way, also our friends and families for their valuable suggestions and support.

**Apurv Kamalakar Badave (B190364217)**

**Aniruddha Shivaji Lalge (B190364389)**

**Nikhil Lalasaheb Elajale (B190364283)**

**Pankaj Dayaram Godara (B190364304)**

**ABSTRACT**

Social media platforms have become integral to modern communication, enabling users to connect, share, and engage in various activities. However, the rise of fake profiles on platforms like, Instagram possess significant challenges related to user privacy, security, and trust. This work presents a novel approach to identify and classify fake Instagram profiles using machine learning techniques. The findings of this research contribute to the ongoing efforts to combat the proliferation of fake profiles on Instagram and other social media platforms. By leveraging machine learning techniques and a comprehensive feature set, the proposed model demonstrates promising results in identifying and classifying fake profiles, thereby promoting a safer and more trustworthy online environment. This research opens avenues for further exploration, including the integration of real-time data streams and the adaptation of the model to other social media platforms.

**Keywords:** Profile identification, User authentication, Data preprocessing, Model training, Online security, Machine learning**.**

# INDEX

1. Introduction 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1.1 | Background | . . . . . . . . . . . . . . . . . . . . . . . . . . . | 2 |
| 1.2 | Motivation | . . . . . . . . . . . . . . . . . . . . . . . . | 2 |

[1.3 Area Of Project . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2](#_TOC_250018)

[1.4 Technical Keywords . . . . . . . . . . . . . . . . . . . . . . . . . . 3](#_TOC_250017)

[1.5 Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3](#_TOC_250016)

1. Literature Survey 4

[2.1 Study Of Research Paper . . . . . . . . . . . . . . . . . . . . . . . 5](#_TOC_250015)

1. Software Requirement Specification 13
   * 1. [Problem Statement 14](#_TOC_250014)
     2. [Assumptions and Dependencies 14](#_TOC_250013)
   1. [FUNCTIONAL REQUIREMENTS 14](#_TOC_250012)
      1. [System Feature1(Functional Requirement) 14](#_TOC_250011)
      2. [System Feature2(Functional Requirement) 15](#_TOC_250010)
   2. [EXTERNAL INTERFACE REQUIREMENT 15](#_TOC_250009)
      1. [User Interface 15](#_TOC_250008)
      2. [Hardware Interfaces 15](#_TOC_250007)
      3. [Software Interfaces 16](#_TOC_250006)
   3. [NON FUNCTIONAL REQUIREMENT 16](#_TOC_250005)
      1. [Performance Requirements 16](#_TOC_250004)
      2. [Safety Requirement 16](#_TOC_250003)
      3. [Software Quality Attributes 16](#_TOC_250002)
   4. [SYSTEM REQUIREMENTS 17](#_TOC_250001)
      1. [Database Requirements 17](#_TOC_250000)
      2. Software Requirements(Platform Choice) 17
      3. Hardware Requirements 17
   5. ANALYSIS MODEL: SDLC MODEL TO BE APPLIED 18
   6. SYSTEM IMPLEMENTATION PLAN 19

### System Design 20

* 1. SYSTEM ARCHITECTURE 21
     1. Module 22
     2. Data Flow Diagram 22
  2. UML DIAGRAMS 24

### Software Information 27

### 5.1 Python………………………………………………………………28

### 5.2 Anaconda……………………………………………………………29

5.3 SPYDER 31

* + 1. Features 31
    2. Other Specifications 32

### Software Testing 33

* 1. Introduction 34
  2. TYPES OF TESTING USED 34
     1. Unit Testing 34
     2. Integration Testing 34
     3. White Box & Black Box Testing 35
  3. Test Cases & Test Resut 36

### Result 38

* 1. Login Page 39
  2. Registration Page 39
  3. GUI Main 40
  4. Outputs 41

1. Conclusion 45
2. References 47
3. Appendix 50
   1. Appendix A 50
   2. Appendix B 51
   3. Appendix C 66

# List of Figures

* 1. System Architecture 21
  2. Data Flow(0) diagram 23
  3. Data Flow(1) diagram 23
  4. Data Flow(2) diagram 23
  5. Class Diagram 24
  6. Usecase Diagram 25
  7. Activity Diagram 25
  8. Sequence Diagram 26

6.1 GUI Testing 36

6.2 Registration Test Case 36

6.3 Login Test Case 37

7.1 Login Page 39

7.2 Registration Page 40

7.3 GUI Main 41

7.4.1 Output 1 Page 42

7..4.1 Output 2 Page 42

7.4.2 Output 3 Page 43

7.4.2 Output 4 Page 44

7.4.2 Output 5 Page 44

**CHAPTER 1**

**INTRODUCTION**

* 1. **BACKGROUND**

Fake Instagram profiles can range from automated bots posting spam to sophisticated imposters attempting to deceive genuine users for financial gain, social manipulation, or other illicit activities. Traditional methods of manual inspection and reporting are insufficient to handle the sheer volume of profiles and interactions, necessitating the use of advanced technological solutions. Machine learning has emerged as a powerful tool in addressing the issue of fake profiles on social media platforms. By harnessing the computational power of machine learning algorithms, it is possible to automatically identify and classify fake profiles based on distinctive patterns and characteristics. The combination of the growing influence of social media, the challenges posed by fake profiles, and the advancements in machine learning techniques has led to the development of solutions aimed at identifying and classifying these profiles. This research addresses the need for a safer and more trustworthy online environment by proposing a comprehensive approach to tackle the issue of fake Instagram profiles using machine learning.

## 1.2 MOTIVATION

The research on “Fake Instagram Profile Identification and Classification using Machine Learning” is relevant due to its potential to address critical issues related to user trust, online safety, and platform integrity. By leveraging the power of machine learning, this research offers practical solutions that align with the needs of the digital age.

### 1.3 AREA OF PROJECT

1. Machine Learning

### 

### 1.4 TECHNICAL KEYWORDS

1. Machine Learning
2. SVM/RF/DT Algorithm

3. Pre-processing

4. Feature Extraction

### 1.5 OBJECTIVES

The research aims to contribute to a safer online environment with enhanced user experiences and support the ongoing efforts of social media platforms to combat fake profiles.

Identify and extract relevant features from the collected Instagram profiles and associated content. The research encompass developing an effective machine learning based model for identifying and classifying fake Instagram profiles.

# CHAPTER 2

# LITERATURE SURVEY

The research encompass developing an effective machine learning-based model for identifying and classifying fake Instagram profiles.

### STUDY OF RESEARCH PAPER

1. **Paper Name:** PREDICTION OF FAKE INSTAGRAM PROFILES USING MACHINE LEARNING

**Author:** Anupriya1, V. Sowmiya, Dr. G. Devika.

### Abstract:

The majority of people now use social networking sites as part of their everyday lives. Every day, a vast number of people build profiles on social networking sites and connect with others, regardless of their place or time. False identities play an important role in advanced persisted threats and are also involved in other malicious activities. Users of social networking sites not only profit from them, but they also face security concerns about their personal details. To assess who is promoting threats in social networks, we must first identify the user’s social network profiles. It is necessary to differentiate between genuine and fake accounts on social media based on the classification. Detecting fake accounts on social media has historically focused on a number of classification methods. However, it is possible to boost the accuracy of fake profile identification in social media. Machine Learning and technology is used in the proposed work to increase the percentage of fake profile prediction.

1. **Paper Name:** DETECTION OF FAKE ACCOUNTS IN INSTAGRAM USING MACHINE LEARNING.

**Author:** Ananya Dev, Hamsashree Reddy, Manjistha Sinha.

### Abstract:

With the advent of the Internet and social media, while hundreds of people have benefits from the vast sources of information available, there has been an enormous increase in the rise of cybercrimes, particularly targeted towards women. According to a 2019 report in the [4] Economics Times, India has witnessed a 457 rise in cybercrime in the five year span between 2011 and 2016. Most speculate that this is due to impact of social media such as Facebook, Instagram and Twitter on our daily lives. While these definitely help in creating a sound social network, creation of user accounts in these sites usually needs just an email id. A real life person can create multiple fake IDs and hence impostors can easily be made. Unlike the real world scenario where multiple rules and regulations are imposed to identify oneself in a unique manner (for example while issuing one’s passport or driver’s license), in the virtual world of social media, admission does not require any such checks. In this paper, we study the different accounts of Instagram, in particular and try to assess an account as fake or real using Machine Learning techniques namely Logistic Regression and Random Forest Algorithm.

1. **Paper Name:** SURVEY ON FAKE PROFILE DETECTION ON SOCIAL

SITES BY USING MACHINE LEARNING ALGORITHM

**Author:** Kumud Patel, Sudhanshu Agrahari, Saijshree Srivastava

### Abstract:

To avoid the spam message, malicious and cyber bullies activities which are mostly done by the fake profile. These activities challenge the privacy policies of the social network communities. These fake profiles are responsible for spread false informa- tion on social communities. To identify the fake profile, duplicate, spam and bots account there is much research work done in this area. By using a machine-learning algorithm, most of the fake accounts detected successfully. This paper represents the review of Fake Profile Detection on Social Site by Using Machine Learning.

1. **Paper Name:** FAKE ACCOUNTS DETECTION ON SOCIAL MEDIA (INSTAGRAM AND TWITTER)

**Author:** Dr.P.V. Kumar, Shanthi Vardhan, Y. Kavya, K. Badri Singh.

**Abstract:**

Online Social Networks (OSNs) have grown in popularity among today’s youth, having an effect on their social life and motivating them to sign up for various social media platforms. Social media sites offer the required tools for a range of tasks, including news generation, Fake accounts have grown to be a serious issue with the growth of social media, endangering user security and platform integrity. In this work, we investigate how well machine learning (ML) algorithms identify fake accounts on social media sites like Twitter and Instagram. In order to train ML models for spotting fake accounts, we examine user behavior and account attributes, extracting parameters like the number of followers, activity level, and posting behavior. To preprocess the data and use different ML techniques, such Random Forest, Support Vector Machines, and XG boost, to categorize and identify bogus ac- counts, we employ Python packages. The findings demonstrate that ML algorithms can accurately detect patterns and abnormalities suggestive of fake accounts and achieve high precision in fake account detection.

1. **Paper Name:** INSTAGRAM FAKE AND AUTOMATED ACCOUNT DETECTION

**Author:** Fatih Cagatay Akyon, M. Esat Kalfaoglu.

### Abstract:

Fake engagement is one of the significant problems in Online Social Networks (OSNs) which is used to increase the popularity of an account in an inorganic manner. The detection of fake engagement is crucial because it leads to loss of money for businesses, wrong audience targeting in advertising, wrong product predictions systems, and unhealthy social network environment. This study is related with the detection of fake and automated accounts which leads to fake engagement on Instagram. As far as we know, there is no publicly available dataset for fake and automated accounts. For this purpose, two dataset have been generated for the detection of fake and automated accounts. For the detection of these accounts, machine learning algorithms like Naive Bayes, logistic regression, support vector machines and neural networks are applied. Additionally, for the detection of automated accounts, cost sensitive genetic algorithm is applied because of the unnatural bias in the dataset. To deal with the unevenness problem in the fake dataset, Smotenc algorithm is implemented. For the automated and fake account detection problem, 86 and 96 are obtained, respectively.

1. **Paper Name:** Fake Profile Detection Using Machine Learning Techniques

**Author:** Partha Chakraborty, Mahim Musharof Shazan, Mahamudul Nahid, Md. Kaysar Ahmed, Prince Chandra Talukder

### Abstract:

Our lives are significantly impacted by social media platforms such as Facebook, Twitter, Instagram, LinkedIn, and others. People are actively participating in it the world over. However, it also has to deal with the issue of bogus profiles. False accounts are frequently created by humans, bots, or computers. They are used to disseminate rumors and engage in illicit activities like identity theft and phishing. So, in this project, the author’ll talk about a detection model that uses a variety of machine learning techniques to distinguish between fake and real Twitter profiles based on attributes like follower and friend counts, status updates, and more. The author used the dataset of Twitter profiles, separating real accounts into TFP and E13 and false accounts into INT, TWT, and FSF. Here, the author discusses LSTM, XG Boost, Random Forest, and Neural Networks. The key characteristics are chosen to assess a social media profile’s authenticity. Hyperparameters and the architecture are also covered. Finally, results are produced after training the models. The output is therefore 0 for genuine profiles and 1 for false profiles. When a phony profile is discovered, it can be disabled or destroyed so that cyber security problems can be prevented.

1. **Paper Name:** PREDICTION OF FAKE INSTAGRAM PROFILES USING MA CHINE LEARNING

**Author:** S. Saranya Shree, C. Subhiksha, R. Subhashini

### Abstract:

The majority of people now use social networking sites as part of their everyday lives. Every day, a vast number of people build profiles on social networking sites and connect with others, regardless of their place or time. Users of social networking sites not only profit from them, but they also face security concerns about their personal details. To assess who is promoting threats in social networks, we must first identify the users’ social network profiles. We may differentiate between genuine and false accounts on social media based on the classification. Detecting false ac- counts on social media has historically focused on a number of classification methods. However, we must boost the accuracy of fake profile identification in social media. We suggest machine learning and natural language processing (NLP) in this paper to increase the percentage of fake profile prediction. The Support Vector Ma- chine (SVM) and the Naive Bayes algorithm are two algorithms that can be used.

1. **Paper Name:** Fake Profile Identification using Machine Learning

**Author:** Samala Durga, Prasad Reddy

### Abstract:

In the present generation, the social life of everyone has become associated with on- line social networks. These sites have made a drastic change in the way we pursue our social life. Making friends and keeping in contact with them and their updates has become easier. But with their rapid growth, many problems like fake profiles, on- line impersonation have also grown. There are no feasible solutions exist to control these problems. In this paper, I came up with a framework with which the automatic identification of fake profiles is possible and is efficient. This framework uses classification techniques like Random Forest Classifier to classify the profiles into fake or genuine classes. As this is an automatic detection method, it can be applied easily by online social networks that have millions of profiles whose profiles cannot be examined manually.

# 

# CHAPTER 3

# SOFTWARE REQUIREMENT

# SPECIFICATION

### PROBLEM STATEMENT

To develop a windows based model to identify and classify fake Instagram profiles using machine learning algorithms such as SVM, Random Forest and Decision Tree algorithms.

### ASSUMPTIONS AND DEPENDENCIES

1. User must install the Python on his pc.
2. User has to install the Spyder on his pc.
3. User has to login to the system.

### FUNCTIONAL REQUIREMENTS

### System Feature1 (Functional Requirement)

**Admin:** Admin module will be on web module. Admin will verify user information and allow or reject to user. Load the Data set.

**User:** User registers into system with personal information. Automatically user verification request send to admin. After verification user can login into system.

### System Feature2 (Functional Requirement)

**System:** By using RF algorithm, enhance Machine Learning techniques like Feature Selection, Handling Missing Data & Improving Stability and Accuracy.

### 

### EXTERNAL INTERFACE REQUIREMENT

### User Interface

Application of fake Instagram Profile Identification and Classification using Machine Learning.

### Hardware Interfaces:

* + RAM: 8GB

As we are using Machine Learning Algorithm, various high level Libraries and data loading should be fast hence minimum laptop RAM required is 8 GB.

* + Hard Disk: 40 GB
  + Processor: Intel i5 Processor
  + IDE: Spyder

Spyder IDE the Integrated Development Environment to be used.

* + Coding Language: Python Version 3.5

Highly specified programming language for Machine Learning because of availability of High Performance Libraries.

* + Operating System: Windows 11

### Software Interfaces

* + Operating System: Windows 11
  + IDE: Spyder
  + Programming Language: Python

### NON FUNCTIONAL REQUIREMENT

### Performance Requirements

The performance of the functions and every module must be well. The overall performance of the software will enable the users to work decently. Performance of encryption of data should be fast. Performance of the providing virtual environment should be fast.

### Safety Requirement

The application is designed in modules where errors can be detected and fixed easily. This makes it easier to install and update new functionality if required.

### Software Quality Attributes

Our software has many quality attributes that are given below:

* Adaptability: This software is adaptable by all users.
* Availability: This software is freely available to all users. The availability of the software is easy for everyone.
* Maintainability: After the deployment of the project if any error occurs then it can be easily maintained by the software developer.
* Reliability: The performance of the software is better which will increase the reliability of the Software.
* User Friendliness: Since, the software is a GUI application; the output generated is much user friendly in its behavior.
* Integrity: Integrity refers to the extent to which access to software or data.
* Security: Users are authenticated using many security phases so reliable security is provided.
* Testability: The software will be tested considering all the aspects.

### SYSTEM REQUIREMENTS

### Database Requirements

SQLITE

### Software Requirements (Platform Choice)

* Operating system: Windows 7 or more.
* Coding Language: Python
* IDE: Spyder

### Hardware Requirements

* System: Intel I3 Processor and above.
* Hard Disk: 20 GB
* RAM: 8GB

### ANALYSIS MODEL: SDLC MODEL TO BE APPLIED

The software development cycle is a combination of different phases such as designing, implementing and deploying the project. These different phases of the software development model are described in this section. The SDLC model for the project development can be understood using the following figure. The chosen SDLC model is the waterfall model which is easy to follow and fits bests for the implementation of this project.

**Requirements Analysis:** At this stage, the business requirements, definitions of usecases are studied and respective documentations are generated.

**Design:** In this stage, the designs of the data models will be defined and different data preparation and analysis will be carried out.

**Implementation:** The actual development of the model will be carried out in this stage. Based on the data model designs and requirements from previous stages, appropriate algorithms, mathematical models and design patterns will be used to develop the agent’s backend and front-end components.

**Testing:** The developed model based on the previous stages will be tested in this stage. Various validation tests will be carried out over the trained model.

**Deployment:** After the model is validated for its accuracy scores its ready to be deployed or used in simulated scenarios.

**Maintenance:** During the use of the developed solution various inputs/scenarios will been countered by the model which might affect the models overall accuracy. Or with passing time the model might not fit the new business requirements. Thus, the model must be maintained often to keep its desired state of operation.

### SYSTEM IMPLEMENTATION PLAN

The System Implementation plan table, shows the overall schedule of tasks compilation and time duration required for each task.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Name/Title** | **Start Date** | **End Date** |
| 1 | Preliminary Survey | 22/08/2023 | 29/08/2023 |
| 2 | Introduction and Problem  Statement | 12/09/2023 | 26/09/2023 |
| 3 | Literature Survey | 04/10/2023 | 10/10/2023 |
| 4 | Software Requirement And Specification | 11/10/2023 | 18/10/2023 |
| 5 | System Design | 19/10/2023 | 02/11/2023 |
| 6 | Partial Report Submission | 03/11/2023 | 07/11/2023 |
| 7 | Architecture Design | 19/01/2024 | 08/02/2024 |
| 8 | Implementation | 19/02/2024 | 03/03/2024 |
| 9 | Deployment | 10/03/2024 | 20/03/2024 |
| 10 | Testing | 01/04/2024 | 15/04/2024 |
| 11 | Paper Publish | 16/04/2024 | 21/04/2024 |
| 12 | Report Submission | 24/04/2024 | 03/05/2024 |

# 

# CHAPTER 4

# SYSTEM DESIGN

### SYSTEM ARCHITECTURE:

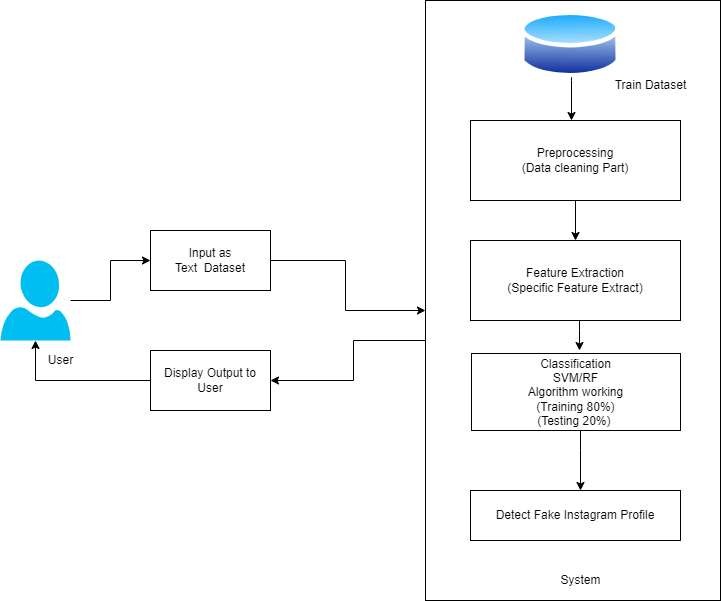


Figure 4.1: System Architecture

### Module

* + - * **Admin**

In this module, the Admin has to log in by using valid user name and password. After login successful he can do some operations such as View All Users and authorize.

* + - * **View and Authorize Users**

In this module, the admin can view the list of users who all registered. In this, the admin can view the user’s details such as, username, email, address and admin authorizes the users.

* + - * **View Charts Results**

View All Products Search Ratio, View All Keywords, Search Results, View All Product Review Rank Results.

* + - * **End User**

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will best or to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like Manage Account.

### Data Flow Diagram

In Data Flow Diagram, we show that flow of data in our system in DFD0 we show that base DFD in which rectangle presents input as well as output and circle shows our system.

In DFD1 we show actual input and actual output of system input of our system is text or image and output is rumor detected likewise in DFD 2 we present operation of user as well as admin.



Figure 4.2: Data Flow (0) diagram

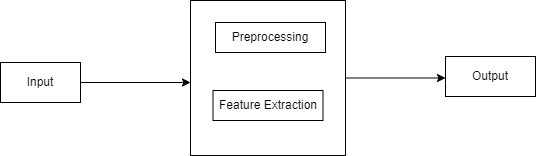


Figure 4.3: Data Flow (1) diagram

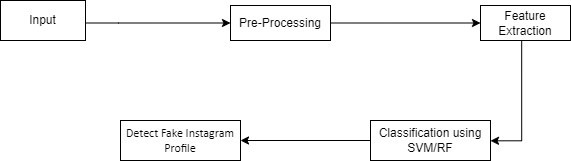


Figure 4.4: Data Flow (2) diagram

### UML DIAGRAMS

Unified Modeling Language is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct and document the artifacts of a software intensive system. UML is process independent, although optimally it should be used in process that is use case driven, architecture-centric, iterative and incremental. The Numbers of UML Diagram are available:

* Class Diagram
* Use case Diagram
* Activity Diagram
* Sequence Diagram

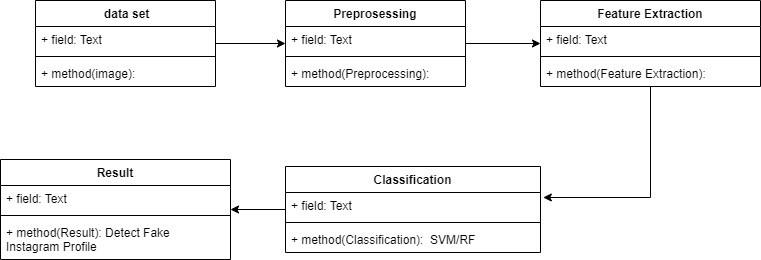


Figure 4.5: Class Diagram

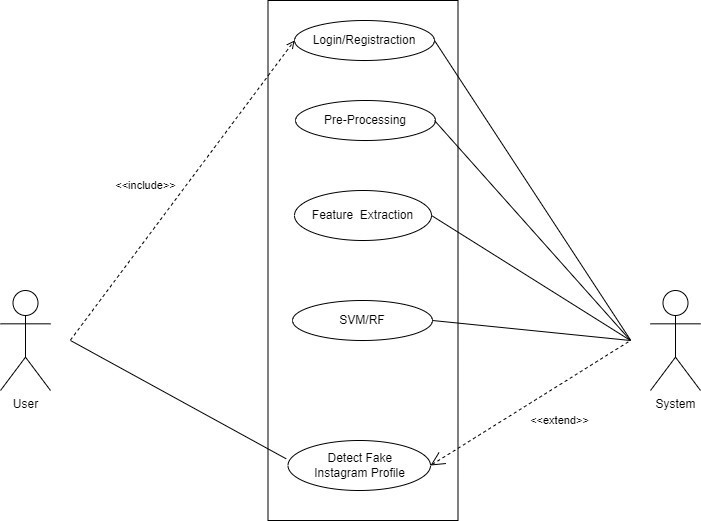


Figure 4.6: Usecase Diagram

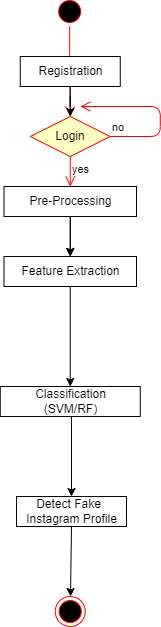


Figure 4.7: Activity Diagram

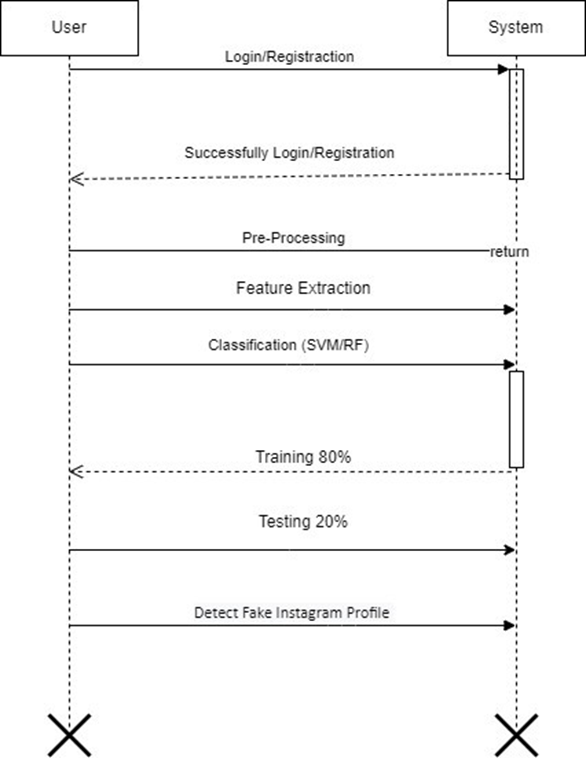


Figure 4.8: Sequence Diagram

# CHAPTER 5

**SOFTWARE INFORMATION**

**5.1 Python**

Python is an interpreted, high-level and general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python’s design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a “batteries included” language due to its comprehensive standard library.

Python was created in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system with reference counting.

Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3.

The Python 2 language was officially discontinued in 2020 (first planned for 2015), and therefore the last Python 2 release.”[30] No more security patches or other improvements will be released for it. With Python 2’s end-of-life, only Python 3.6.x and later are supported.

**5.2 Anaconda**

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012. As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free.

Package versions in Anaconda are managed by the package management system conda. This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called Miniconda, which includes only conda, Python, the packages they depend on, and a small number of other packages.

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPI as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator, as a graphical alternative to the command line interface (CLI). The big difference between conda and the pip package manager is in how package dependencies are managed, which is a significant challenge for Python data science and the reason conda exists.

When pip installs a package, it automatically installs any dependent Python pack- ages without checking if these conflict with previously installed packages. It will install a package and any of its dependencies regardless of the state of the existing installation. Because of this, a user with a working installation of, for example, Google Tensorflow, can find that it stops working having used pip to install a different package that requires a different version of the dependent numpy library than the one used by Tensorflow. In some cases, the package may appear to work but produces different results in detail.

In contrast, conda analyses the current environment including everything currently installed and together with any version limitations specified (e.g. the user may wish to have Tensorflow version 2.0 or higher), works out how to install a compatible set of dependencies, and shows a warning if this cannot be done.

Open source packages can be individually installed from the Anaconda repository, Anaconda Cloud (anaconda.org), or the user’s own private repository or mirror, using the conda install command. Anaconda, Inc. compiles and builds the packages available in the Anaconda repository itself, and provides binaries for Windows 32/64 bit, Linux 64 bit and MacOS 64-bit. Anything available on PyPI may be installed into a conda environment using pip and conda will keep track of what it has installed itself and what pip has installed.

Custom packages can be made using the conda build command, and can be shared with others by uploading them to Anaconda Cloud, PyPI or other repositories.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it is possible to create new environments that include any version of Python packaged with conda.

### 5.3 Spyder

Spyder is a powerful scientific environment written in Python, for Python, and de- signed by and for scientists, engineers and data analysts. It offers a unique combination of the advanced editing, analysis, debugging, and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection, and beautiful visualization capabilities of a scientific package.

Beyond its many built-in features, its abilities can be extended even further via its plugin system and API. Furthermore, Spyder can also be used as a PyQt5 extension library, allowing you to build upon its functionality and embed its components, such as the interactive console, in your own software.

### 5.3.1 Features

* **Editor**

Works efficiently in a multi-language editor with a function/class browser, real- time code analysis tools (pyflakes, pylint, and pycodestyle), automatic code completion (jedi and rope), horizontal/vertical splitting, and go-to-definition.

* **Interactive Console**

Harness the power of as many IPython consoles as you like with full workspace and debugging support, all within the flexibility of a full GUI interface. Instantly run your code by line, cell, or file, and render plots right inline with the output or in interactive windows.

* **Documentation Viewer**

Render documentation in real-time with Sphinx for any class or function, whether external or user-created, from either the Editor or a Console. It is very useful for documenting viewing.

* **Variable Explorer**

Inspect any variables, functions or objects created during your session. Editing and interaction is supported with many common types, including numeric/strings/bools, Python lists/tuples/dictionaries, dates/timedeltas, Numpy arrays, Pandas in- dex/series/dataframes, PIL/Pillow images, and more.

* **Development Tools**

Examine your code with the static analyzer, trace its execution with the interactive debugger, and unleash its performance with the profiler. Keep things organized with project support and a built-in file explorer, and use find in files to search across entire projects with full regex support.

**5.3.2 Other Specifications**

* **Advantages**

Machine learning models can efficiently process large volumes of data, making it feasible to analyze numerous profiles and interactions in real-time. Machine learning enables the automation of fake profile identification, reducing the reliance on manual inspections and user reporting. The proposed approach incorporates both content-based and network-based attributes for classification. Machine learning algorithms are capable of learning intricate patterns that distinguish genuine from fake profiles.

* **Limitations**

The effectiveness of machine learning models heavily relies on the quality and quantity of available data. Developing effective algorithms to capture both content-based and network based attributes accurately requires careful consideration and domain expertise

# CHAPTER 6

# SOFTWARE TESTING

# 6.1 INTRODUCTION

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted. Different software development models will focus the test effort at different points in the development process. Newer development models, such as Agile, often employ test driven development and place an increased portion of the testing in the hands of the developer, before it reaches a formal team of testers.

**6.2 TYPES OF TESTING USED**

Involves various testing strategies used for the project.

**6.2.1 UNIT TESTING**

It is the testing of individual software units of the application. It is done after the complexion of an individual unit before integration. Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. Unit tests perform basic tests at component level and test a specific business process, application, and system configuration.

## INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields.

Integration tests demonstrate that although the components were individually satisfaction, as shown. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

## WHITE-BOX TESTING & BLACK-BOX TESTING

Software testing methods are traditionally divided into white and black-box testing. These two approaches are used to describe the point of view that a test engineer takes while designing test cases:

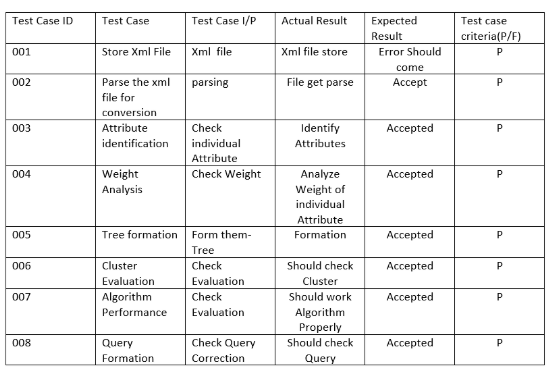
1. **White-Box Testing**

In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases.

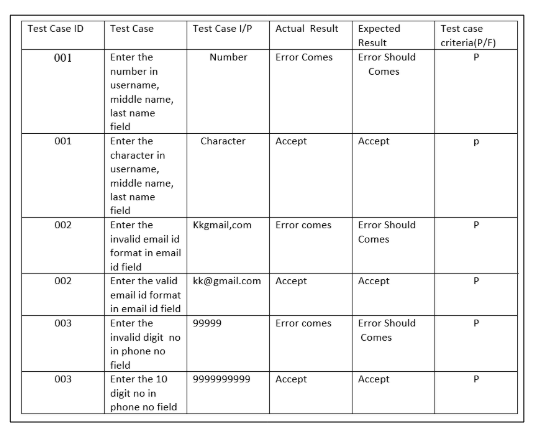
**2. Black-Box Testing**

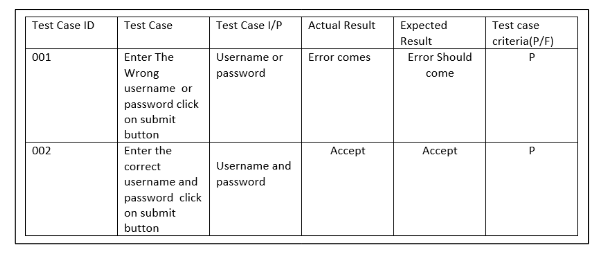
Black-box testing treats the software as a examining functionality without any knowledge of internal implementation. The testers are only aware of what the software is supposed to do, not how it does it.

**6.3 TEST CASES TEST RESULTS**

****

**Figure 6.1: GUI TESTING**

**Figure 6.2: Registration Test Case**

****

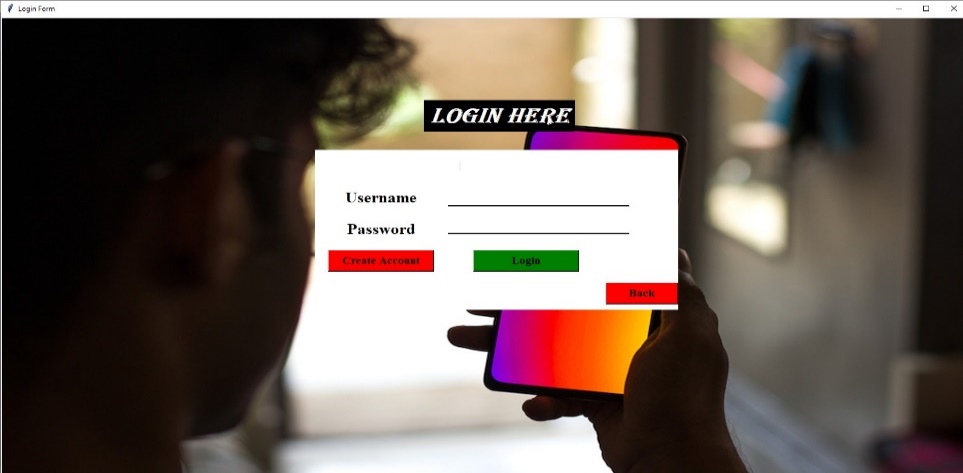
**Figure 6.3: Login Test Case**

# CHAPTER 7

# RESULT

**7.1 Login Page**

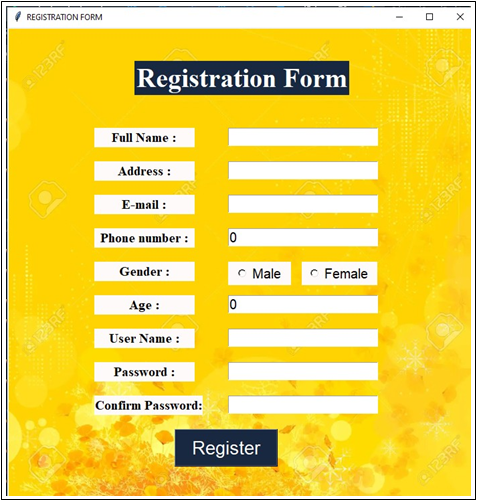
The login page in the project allows users to access the system where machine learning algorithms analyze and classify Instagram profiles, distinguishing between fake and genuine accounts based on various data inputs.



**Figure 7.1: Login Page**

**7.2 Registration Page**

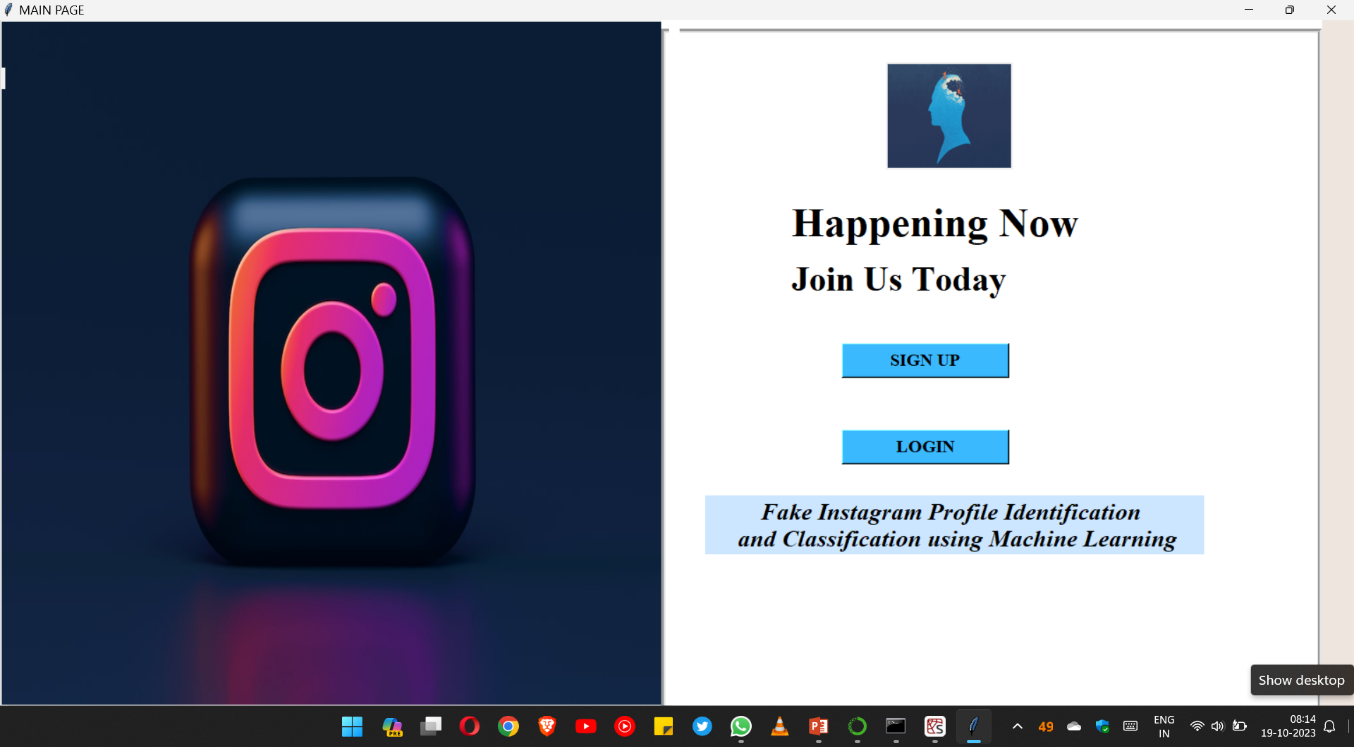
The registration page captures essential user details such as name, email, password, and sometimes additional verification information. This data is used to authenticate users, store their preferences, and secure their interactions within the system. After registration, users can access the machine learning tools that analyze and classify Instagram profiles to detect and identify fake accounts. The process ensures that only authorized users can utilize the system's capabilities, enhancing security and user management.

****

**Figure 7.2: Registration Page**

**7.3 GUI Main**

The GUI Main page is the primary interface of the Fake Instagram Profile Identification and Classification system. It features a user-friendly dashboard where users can Sign Up or Log in. This central hub ensures easy interaction with the system's machine learning features, providing a seamless user experience for effective profile analysis.



**Figure 7.3: GUI Main**

**7.4 Outputs**

**7.4.1 Page1 & Page2**

These pages contain various attributes like followers, following, username etc. of an Instagram profile whose authenticity to be known. Also based upon these attributes and classification using ML algorithms it displays messages like **Account\_Not\_Fake** (if it’s not fake) & **Fake\_Account** (if it’s fake).



**Figure 7.4.1: Output Page1**



**Figure 7.4.1: Output Page2**

**7.4.2 Page3, Page4 & Page5**

These 3 pages include detailed performance metrics for three algorithms: SVM (Support Vector Machine), RF (Random Forest), and DT (Decision Tree). These metrics comprise precision, recall, F-score, and support, displayed to help users evaluate the accuracy and effectiveness of each algorithm. Also there is an option to check authenticity of the profile manually by clicking on “check” button.

Accuracies of the algorithms used:

1. Support Vector Machine: 95.83%
2. Random Forest: 97.22%
3. Decision Tree: 95.83%

# 

**Figure 7.4.2: Output Page3**

# 

**Figure 7.4.2: Output Page4**

# 

**Figure 7.4.2: Output Page5**

# CHAPTER 8

# CONCLUSION

The research on “Fake Instagram Profile Identification and Classification using Machine Learning” presents a comprehensive approach to tackle the persistent issue of fake profiles on social media platforms, with a specific focus on Instagram. By leveraging the power of machine learning techniques, this research contributes to creating a safer and more trustworthy online environment for users, bolstering user confidence, and upholding the integrity of social media community. The research’s outcomes extend beyond the realm of academia, impacting the lives of individuals, businesses, and society as a whole. As social media continues to shape the digital landscape, the work presented here contributes to building a foundation of trust and authenticity, reinforcing the positive potential of online interactions and collaborations.

# CHAPTER 9

# REFERENCES

1. Aleksei Romanov, Alexander Semenov, Oleksiy Mazhelis and Jari Veijalainen.2017. ”Detection of Fake Profiles in Social Media”. In 13th International Conference

on Web Information Systems and Technologies.

1. Indira Sen,Anupama Aggarwal,Shiven Mian.2018.”Worth its Weight in Likes: Towards Detecting Fake Likes on Instagram”. In ACM International Con- ference on Information and Knowledge Management.Nazir, Atif, Saqib Raza, Chen-Nee Chuah, Burkhard Schipper, and C. A. Davis. “Ghostbusting Face- book: Detecting and Characterizing Phantom Profiles in Online Social Gam- ing Applications.” In WOSN. 2010.
2. Nambouri Sravya, Chavana Sai praneetha, S. Saraswathi,” Identify the Human or Bots Twitter Data using Machine Learning Algorithms”, International Re- search Journal of Engineering and Technology (IRJET), Volume: 06 Issue: 03Mar 2019 [www.irjet.net,](http://www.irjet.net/) e-ISSN: 2395-0056, p- ISSN: 2395-0072.
3. M. Smruthi, N. Harini,” A Hybrid Scheme for Detecting Fake Accounts in Facebook”, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7, Issue-5S3, February 2019.
4. Nazir, Atif, Saqib Raza, Chen-Nee Chuah, Burkhard Schipper, and C. A. Davis. “Ghostbusting Facebook: Detecting and Characterizing Phantom Pro- files in Online Social Gaming Applications.” In WOSN. 2010.
5. Rao, P. S., J. Gyani, and G. Narsimha. ”Fake profiles identification in online social networks using machine learning and NLP.” Int. J. Appl. Eng. Res 13.6 (2018): 973-4562.
6. Raturi, Rohit. ”Machine learning implementation for identifying fake ac- counts in social network.” International Journal of Pure and Applied Math- ematics 118.20 (2018): 4785-4797. J. Wang, “Fundamentals of erbium-doped fibre amplifiers arrays (Periodical style—Submitted for publication),” IEEE J. Quantum Electron., submitted for publication.
7. M. Mohammadrezaei, M. E. Shiri, and A. M. Rahmani, “Identifying fake ac- counts on social networks based on graph analysis and classification algo- rithms,” Security and Communication Networks, vol. 2018, 2018.
8. Ala’M, Al-Zoubi, Ja’far Alqatawna, and Hossam Faris. ”Spam profile detec- tion in social networks based on public features.” 2017 8th International Con- ference on information and Communication Systems (ICICS). IEEE, 2017.
9. Romanov, Aleksei, Alexander Semenov, Oleksiy Mazhelis, and Jari Veijalainen. ”Detection of fake profiles in social media-Literature review.” In International Conference on Web Information Systems and Technologies, vol. 2, pp. 363-369. SCITEPRESS, 2018.

**APPENDIX**

**APPENDIX A**

**Type of Problem: NP (Non Deterministic Polynomial Time) Class**

**Definition:** Problems for which a proposed solution can be verified quickly in polynomial time by a deterministic approach are called as NP Class problems.

**Explanation:**

The problem of Fake Instagram Profile Identification and Classification using Machine Learning can be categorized as an NP (Nondeterministic Polynomial time) class problem for the following reasons:

**1. Verification in Polynomial Time:**

Once a machine learning model (e.g., SVM, Random Forest, Decision Tree) is trained, it produces a classification for each Instagram profile (either fake or genuine).

**2. Decision Problem Aspect:**

For any given instance (profile and model), we can quickly verify if the model's decision is correct by comparing it with the ground truth. This quick verification aligns with the NP class definition.

**3. Non-Deterministic Polynomial Time:**

Imagine a hypothetical non-deterministic machine that guesses the classification (fake or genuine) for each profile. We could then verify these guesses in polynomial time by comparing them against known labels.

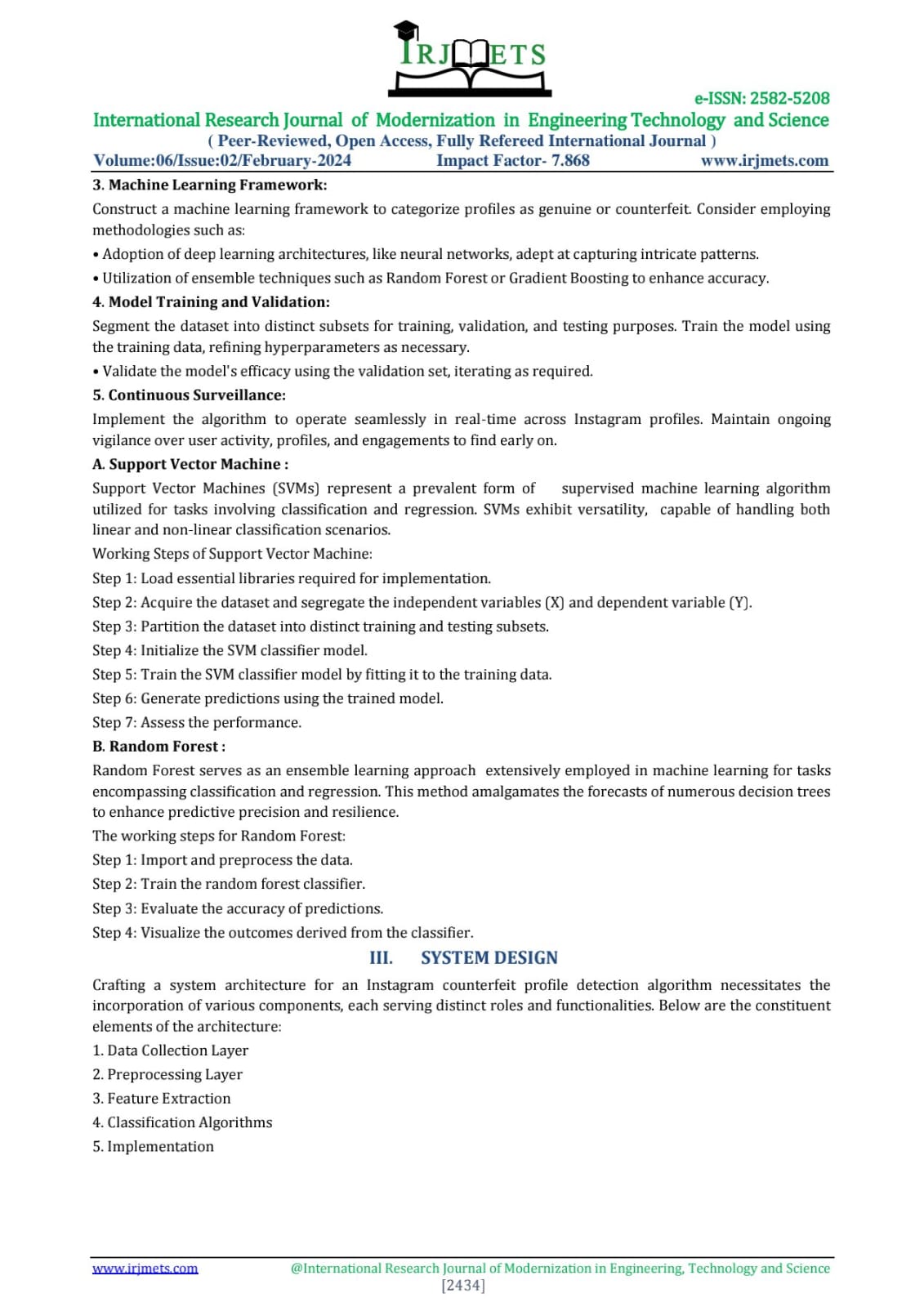
**APPENDIX B**

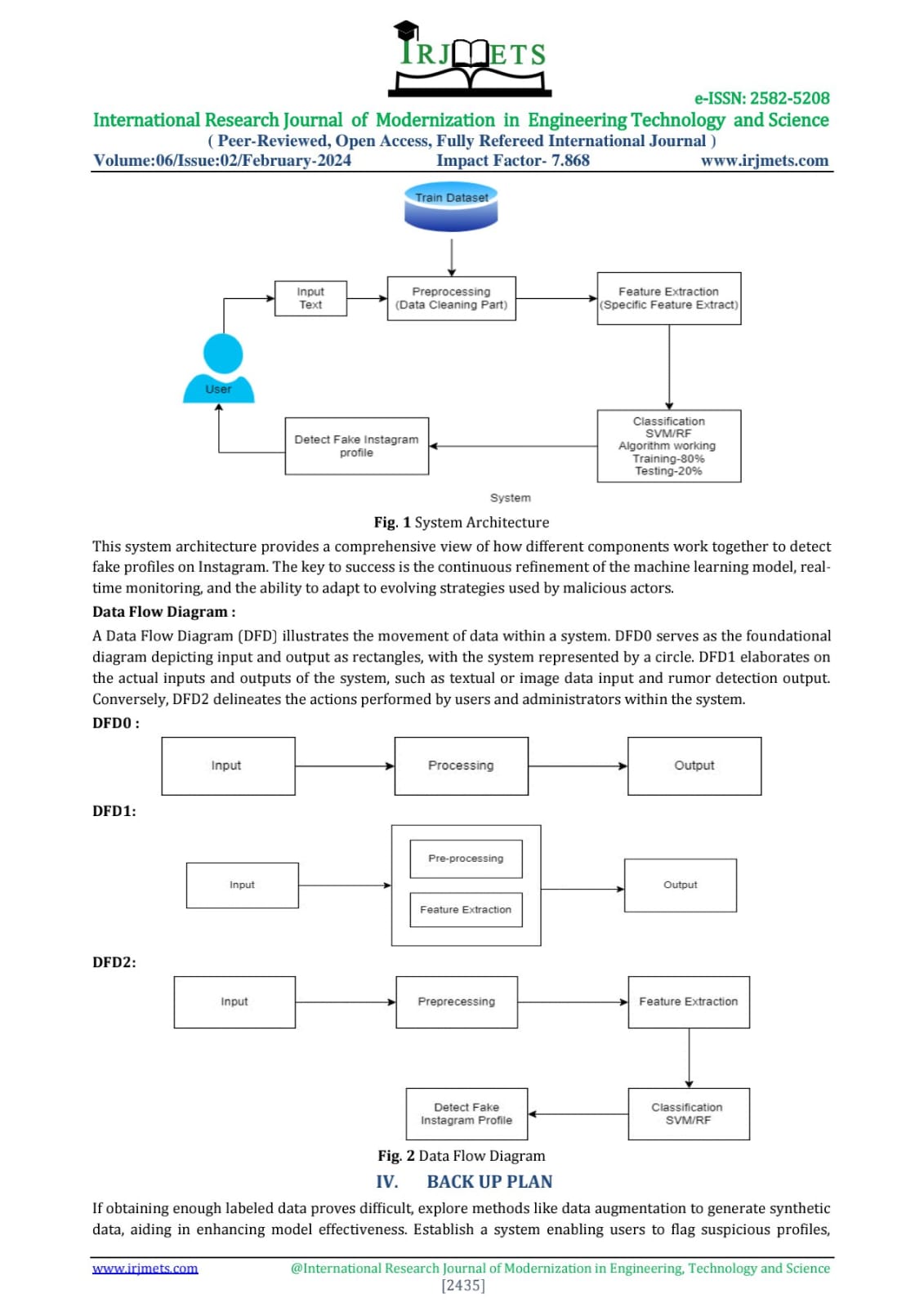
**1. Paper Name:** **FAKE INSTAGRAM PROFILE IDENTIFICATION**

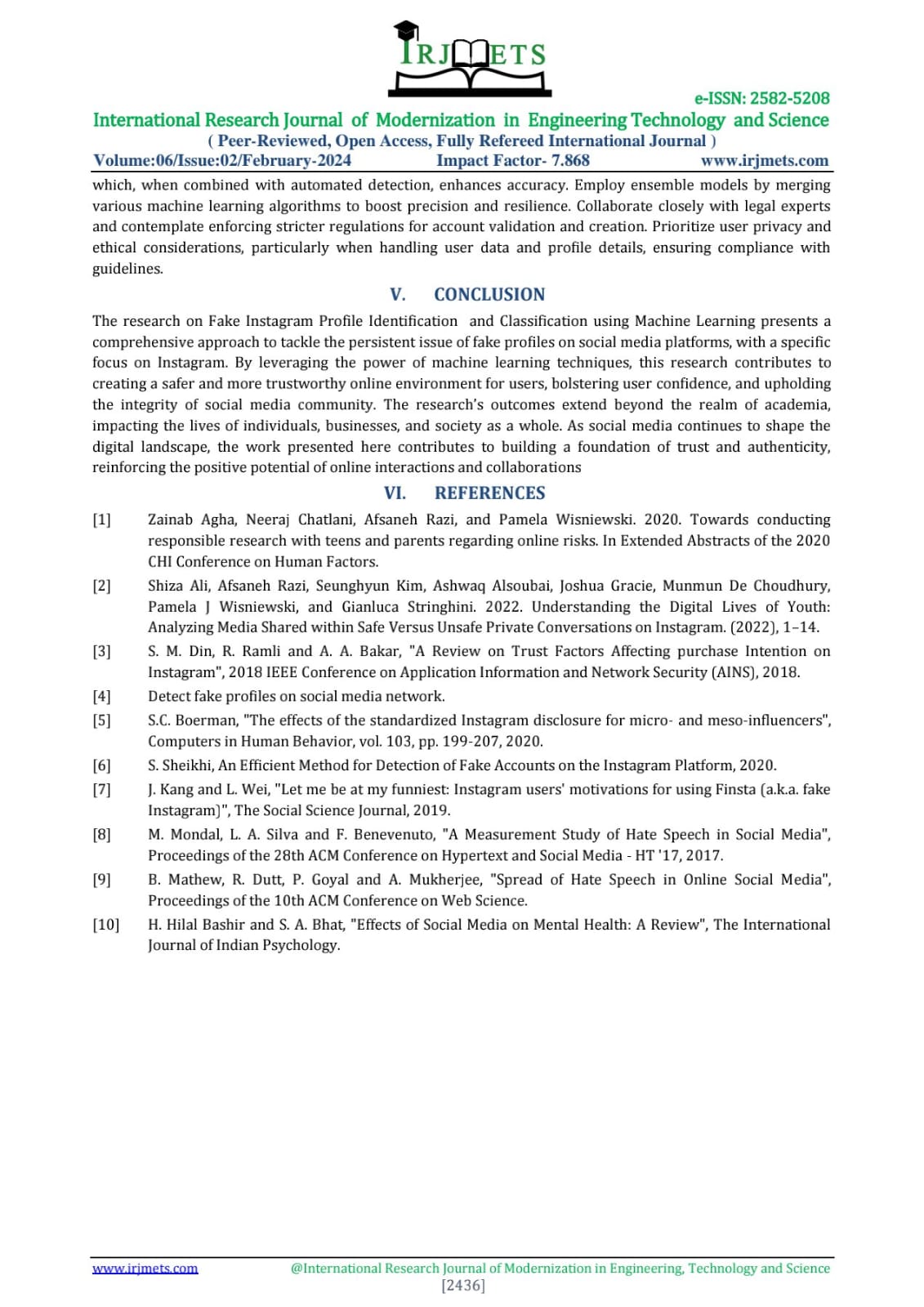
**Journal Name:** **International Research Journal of Modernization in Engineering Technology and Science**

**Research Paper :**









**Certificates:**



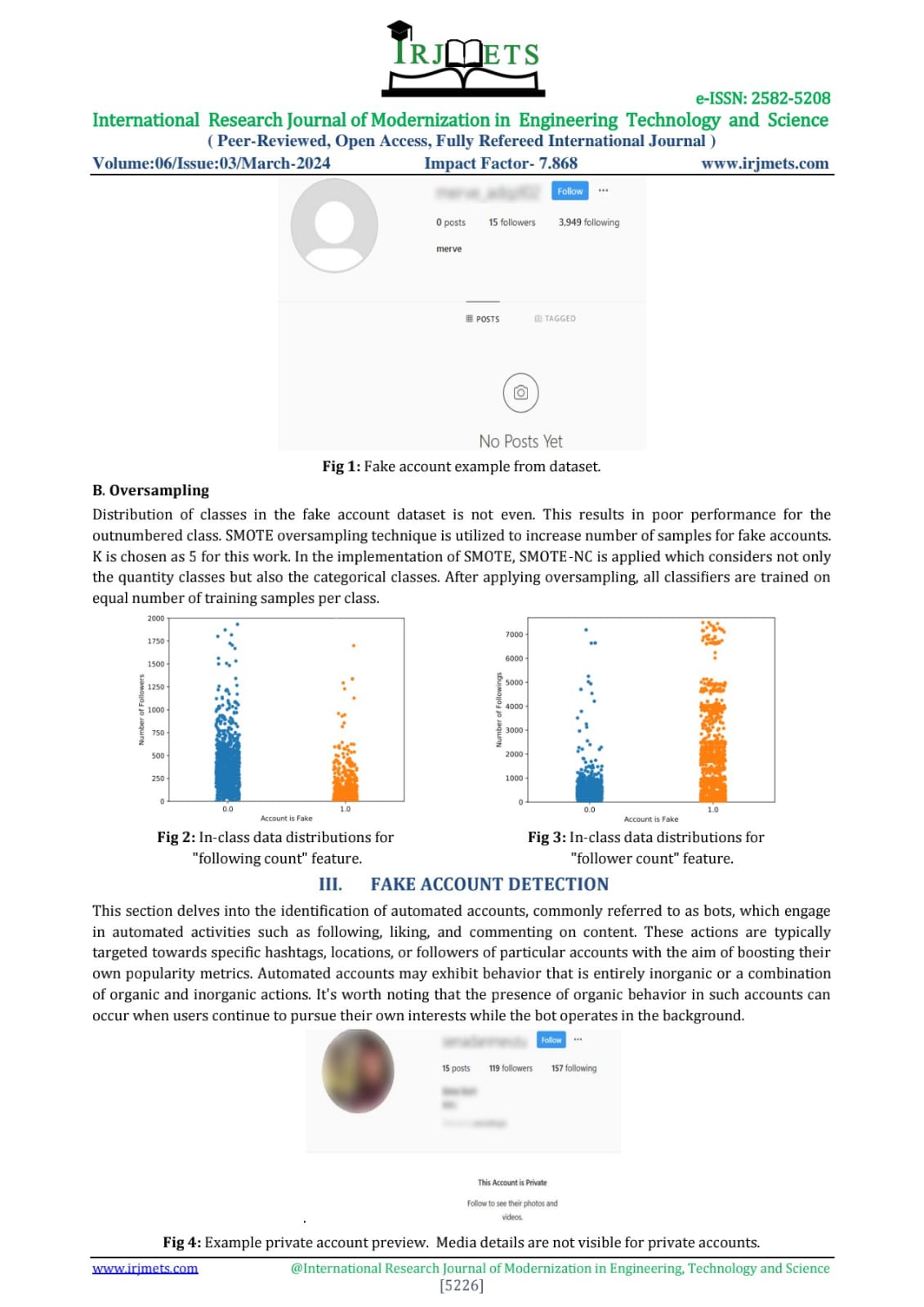


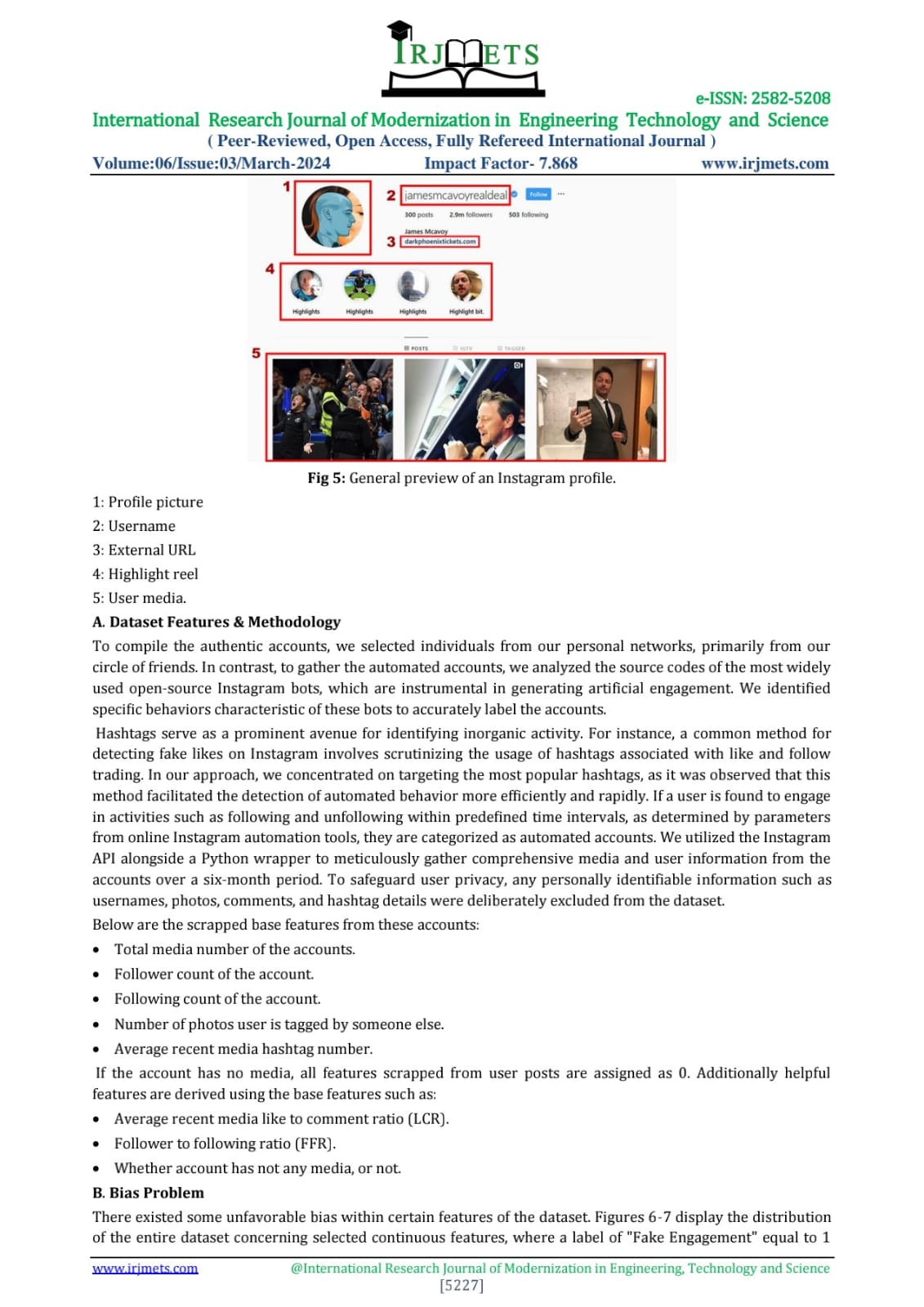
**2. Paper Name:** **FAKE INSTAGRAM PROFILE IDENTIFICATION & CLASSIFICATION USING ML**

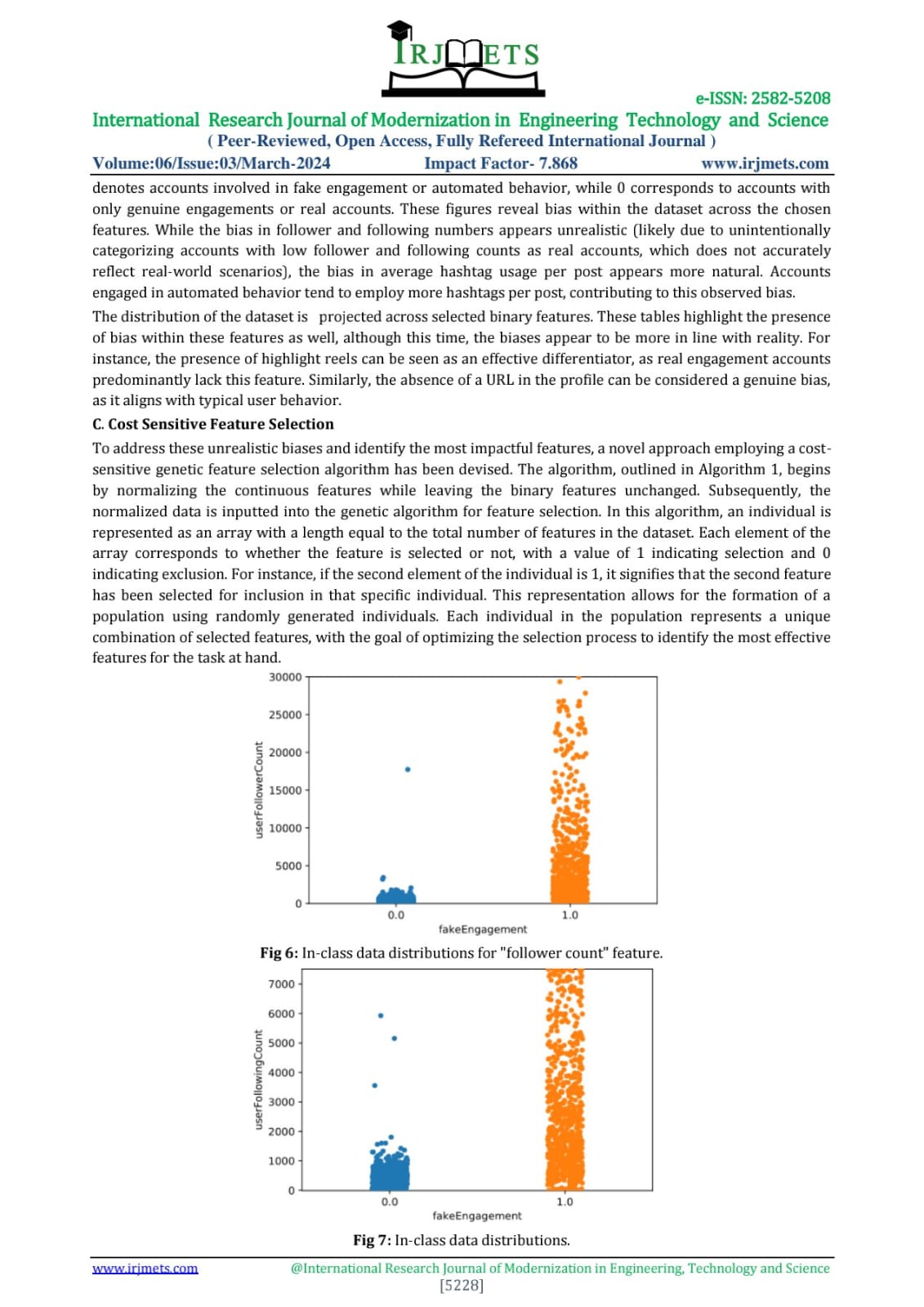
**Journal Name:** **International Research Journal of Modernization in Engineering Technology and Science**

**Research Paper :**



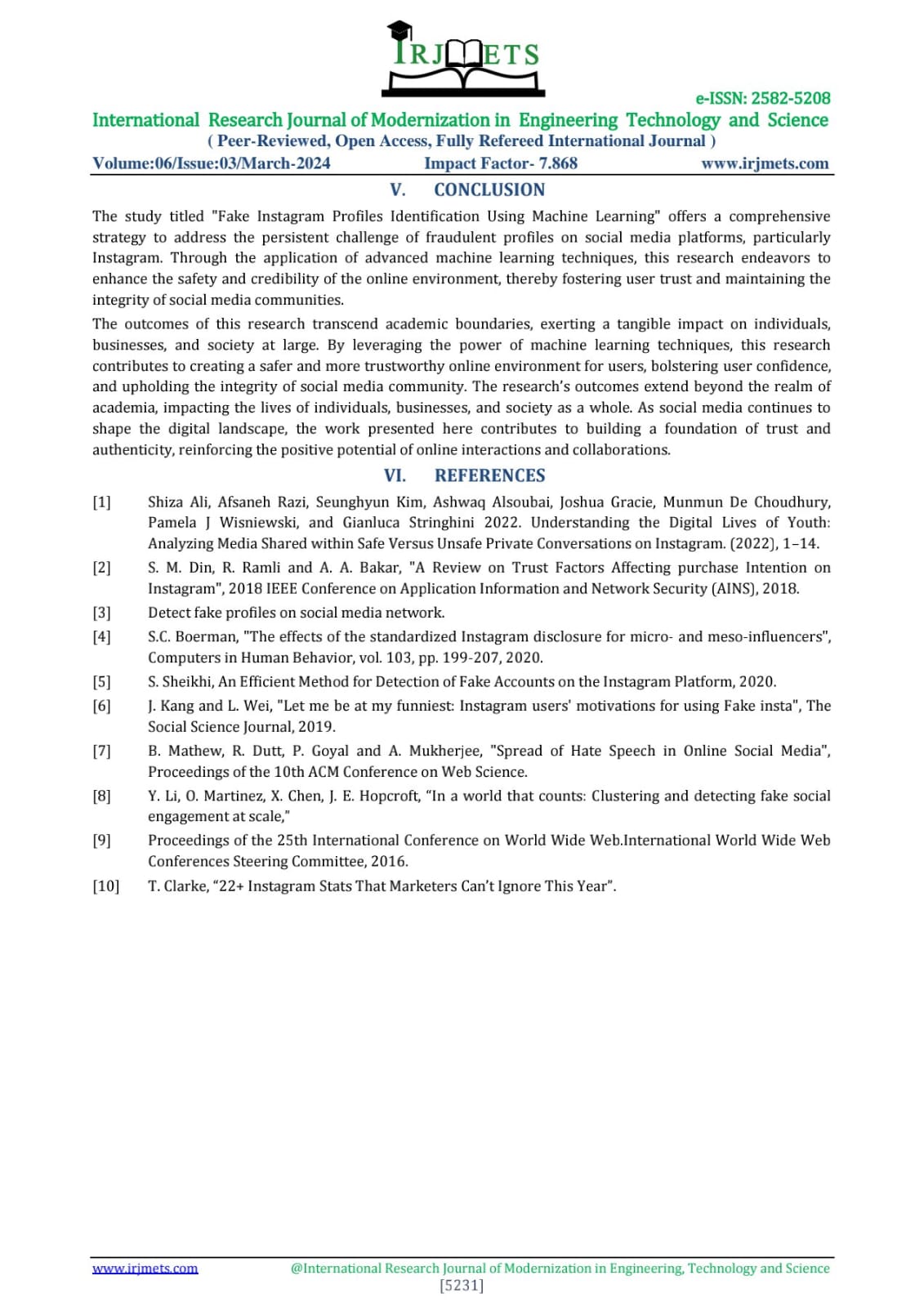












**Certificates:**





**APPENDIX C**

**Plagiarism Report:** **Plagiarism Report of Published Paper.**

