A

Major Project On

## FAKE PROFILE IDENTIFICATION IN SOCIAL NETWORK

**USING MACHINE LEARNING AND NLP**

(Submitted in partial fulfillment of the requirements for the award of Degree) BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

KOPPERA NIKITHA REDDY (207R1A05F0) BATHINI ABHIRAM (207R1A05C9)

JACINTH PHILIP (207R1A05E4)

Under the Guidance of

**K. SHILPA**

(Assistant Professor)



### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**CMR TECHNICAL CAMPUS**

**UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, NewDelhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road,

Hyderabad-501401.

**2020-2024**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



## CERTIFICATE

This is to certify that the project entitled **“FAKE PROFILE IDENTIFICATION IN SOCIAL NETWORK USING MACHINE LEARNING AND NLP ”** being submitted by **KOPPERA NIKITHA REDDY (207R1A05F0), B. ABHIRAM (207R1A05C9) JACINTH PHILIP (207R1A05E4)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023- 24.

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

**K. SHILPA Dr. A. Raji Reddy**

(Assistant Professor) DIRECTOR

INTERNAL GUIDE

**Dr. K. Srujan Raju EXTERNAL EXAMINER**

HOD

**Submitted for viva voice Examination held on**

## ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **K.SHILPA**, Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Committee Review (PRC) **Dr. J. Narsimha Rao, G. Vinesh Shankar, Ms. Shilpa, Dr. K. Maheshwari & Saba Sultana** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the courseof this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

**KOPPERA NIKITHA REDDY (207R1A05F0) BATHINI ABHIRAM (207R1A05C9)**

**JACINTH PHILIP (207R1A05E4)**

## ABSTRACT

In the contemporary landscape, social network sites have seamlessly integrated into the daily lives of a substantial portion of the global population. The ubiquitous creation of user profiles and continuous interactions, transcending geographical and temporal constraints, underscores the pervasive nature of these platforms. However, this widespread connectivity also introduces security challenges, necessitating a nuanced approach to discern genuine users from potential threats. This study addresses the imperative need to classify social network profiles, differentiating between authentic and fake accounts. While existing classification methods have laid a foundation, there exists a pressing requirement to elevate the accuracy of fake profile detection. This paper proposes the integration of advanced machine learning and Natural Language Processing (NLP) techniques to augment the precision of this critical task. The utilization of Support Vector Machine (SVM) and Naïve Bayes algorithms stands out as a cornerstone of this approach. Leveraging the power of machine learning, SVM offers robust pattern recognition, while Naïve Bayes, rooted in probabilistic reasoning, adds a layer of sophistication to the classification process. By incorporating these techniques, the aim is to create a more discerning system that can adeptly identify and segregate genuine user profiles from potentially malicious ones. The focus on NLP brings linguistic patterns into the forefront of analysis. Going beyond traditional methods, this approach scrutinizes the intricacies of language usage within profiles, posts, and comments. This linguistic analysis, coupled with machine learning algorithms, contributes to a holistic evaluation that extends beyond behavioral patterns, enriching the classification process. The overarching goal is to elevate the accuracy rate of fake profile detection in social networks, addressing the evolving nature of deceptive practices. The proposed methodology aligns with the dynamic and multifaceted nature of online interactions, aiming to fortify the security of users in an ever-expanding digital social sphere. This research not only contributes to the academic discourse on online security but also holds practical implications for social network administrators and users alike, fostering a safer and more reliable digital ecosystem.

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **FIGURE NAME** | **PAGE NO** |
| Figure 3.1 | Project Architecture for Fake profile Identification in Social Network using Machine Learning and NLP |  |
|  | 7 |
| Figure 3.3 | Use Case diagram for Fake profile Identification in Social Network using Machine Learning and NLP |  |
|  | 9 |
| Figure 3.4 | Class diagram for Fake Profile |  |
|  | Identification in Social Network using Machine Learning and NLP | 10 |
| Figure 3.5 | Sequence diagram for Fake profile Identification in Social Network using Machine Learning and NLP |  |
|  | 11 |
| Figure 3.6 | Activity diagram for Fake profile  Identification in Social Network using Machine Learning and NLP |  |
|  | 12 |

|  |  |  |
| --- | --- | --- |
| **SCREENSHOT NO.** | **SCREENSHOT NAME** | **PAGE NO**. |
| Figure 5.1 | Login | 27 |
| Figure 5.2 | Prediction | 27 |
| Figure 5.3 | Identifying Fake Profiles | 28 |
| Figure 5.4 | Bar Chart | 28 |

## TABLE OF CONTENTS

**PAGE NO**

**ABSTRACT** i

LIST OF FIGURES ii

LIST OF SCREENSHOTS iii

1. [INTRODUCTION](#_TOC_250024)
   1. [PROJECT SCOPE 1](#_TOC_250023)
   2. [PROJECT PURPOSE 1](#_TOC_250022)
   3. [PROJECT FEATURES 1](#_TOC_250021)
2. [SYSTEM ANALYSIS](#_TOC_250020)
   1. [PROBLEM DEFINITION 2](#_TOC_250019)
   2. [EXISTING SYSTEM 2](#_TOC_250018)
      1. LIMITATIONS OF THE EXISTING SYSTEM 3
   3. [PROPOSED SYSTEM 3](#_TOC_250017)
      1. [ADVANTAGES OF PROPOSED SYSTEM 6](#_TOC_250016)
   4. [FEASIBILITY STUDY 7](#_TOC_250015)
      1. ECONOMIC FEASIBILITY 8
      2. [TECHNICAL FEASIBILITY 8](#_TOC_250014)
      3. [SOCIAL FEASIBILITY 8](#_TOC_250013)
   5. [HARDWARE & SOFTWARE REQUIREMENTS 9](#_TOC_250012)
      1. [HARDWARE REQUIREMENTS 9](#_TOC_250011)
      2. [SOFTWARE REQUIREMENTS 9](#_TOC_250010)
3. [ARCHITECTURE](#_TOC_250009)
   1. [PROJECT ARCHITECTURE 10](#_TOC_250008)
   2. [DESCRIPTION 11](#_TOC_250007)
      1. [WEB SERVER 11](#_TOC_250006)
      2. [WEB DATABASE 11](#_TOC_250005)
      3. [SERVICE PROVIDER 11](#_TOC_250004)
      4. REMOTE USER 11
   3. [USE CASE DIAGRAM 12](#_TOC_250002)
   4. [CLASS DIAGRAM 13](#_TOC_250001)
   5. SEQUENCE DIAGRA 14
   6. [ACTIVITY DIAGRAM 16](#_TOC_250000)

**TABLE OF CONTENTS**

1. **IMPLEMENTATION**
   1. SAMPLE CODE 35
2. **RESULTS**
   1. LOGIN 36
   2. PREDICTION 36
   3. IDENTRIFYING FAKE PROFILES 37
   4. BAR CHART 37
3. **TESTING**
   1. INTRODUCTION TO TESTING 38
   2. TYPES OF TESTING 38
      1. UNIT TESTING 38
      2. INTEGRATION TESTING 38
      3. FUNCTIONAL TESTING 39
   3. TEST CASES
      1. CLASSIFICATION 40
4. **CONCLUSION & FUTURE SCOPE**
   1. PROJECT CONCLUSION 41
   2. FUTURE SCOPE 41

**8.REFERENCES**

8.1REFERENCES 42

* 1. GITHUB LINK 43

# INTRODUCTION

## INTRODUCTION

### PROJECT SCOPE

This project ambitiously aims to develop a sophisticated system for the automatic identification of fake profiles within the realm of social networks. Leveraging advanced machine learning and natural language processing techniques, the scope extends to an exhaustive examination of user behavior, linguistic patterns, and multi-modal data integration. The project seeks to provide a comprehensive solution applicable to various social media platforms, enhancing overall security and trustworthiness.

### PROJECT PURPOSE

The primary purpose of this initiative is to proactively address the escalating challenges posed by fake profiles on social networks. By deploying cutting-edge technologies, the project endeavors to fortify the integrity of online communities, mitigating issues such as identity theft, misinformation, and online scams. Ultimately, the purpose is to create a safer and more reliable online environment, fostering trust among users and administrators alike.

### PROJECT FEATURES

Embedded within the project's architecture are features that collectively contribute to its efficacy. The system incorporates sophisticated behavioral analysis, utilizing ML algorithms to scrutinize user behavior for irregular patterns indicative of fake profiles. This includes a nuanced examination of posting frequency, engagement levels, and other online activities.Additionally, the project integrates advanced linguistic pattern recognition through NLP techniques. This involves a deep dive into the linguistic nuances within user-generated content, uncovering anomalies or suspicious language patterns that may signify the presence of a fake profile. The analysis encompasses aspects such as writing style, grammar, and context.

# SYSTEM ANALYSIS

## SYSTEM ANALYSIS

### SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

### PROBLEM DEFINITION

The escalating prevalence of fake profiles on social networks has emerged as a severe threat, compromising the integrity and security of online communities. Identity theft, misinformation, and online scams have become rampant, eroding user trust in these platforms. Current detection methods are manual, time-intensive, and often ineffective against evolving deceptive strategies. The absence of a sophisticated, automated system exacerbates the challenges. This project addresses the pressing need for an advanced solution, utilizing machine learning and natural language processing to comprehensively analyze user behavior, linguistic patterns, and diverse data modalities. By developing an intelligent system capable of proactively identifying and neutralizing fake profiles across various social media platforms, this initiative aims to restore and enhance the security and credibility of online social spaces.

### EXISTING SYSTEM

Chai et al awarded on this paper is a proof-of inspiration gain knowledge of. Even though the prototype approach has employed most effective normal systems in normal language processing and human-pc interplay, the results realized from the user trying out are significant. By using comparing this simple prototype approach with a wholly deployed menu procedure, they've discovered that users, principally beginner users, strongly pick the common language dialog-based approach. They have additionally learned that in an

ecommerce environment sophistication in dialog administration is most important than the potential to manage complex typical language sentences.

In addition, to provide effortless access to knowledge on ecommerce web sites, natural language dialog-based navigation and menu-pushed navigation should be intelligently combined to meet person’s one-of-a-kind wants. Not too long ago, they have got accomplished development of a new iteration of the approach that includes enormous enhancements in language processing, dialog administration and information management. They believed that average language informal interfaces present powerful personalized alternatives to conventional menupushed or search-based interfaces to web sites.

LinkedIn is greatly preferred through the folks who're in the authentic occupations. With the speedy development of social networks, persons are likely to misuse them for unethical and illegal conducts.However, in relation to LinkedIn such behavioral observations are tremendously restrictive in publicly to be had profile data for the customers by the privateness insurance policies. For that reason, there is to conduct distinctive study on deciding on systems for fake profile identification in LinkedIn. Shalinda Adikari and Kaushik Dutta researched and identified the minimal set of profile data that are crucial for picking out false profiles in LinkedIn and labeled the appropriate knowledge mining procedure for such project.

Z. Halim et al. Proposed spatio-temporal mining on social network to determine circle of customers concerned in malicious events with the support of latent semantic analysis. Then compare the results comprised of spatio temporal co incidence with that of original organization/ties with in social network, which could be very encouraging as the organization generated by spatio-temporal co-prevalence and actual one are very nearly each other. Once they set the worth of threshold to right level, we develop the number of nodes i.e. Actor so that they are able to get higher photo. Total, scan indicate that Latent Semantic Indexing participate in very good for picking out malicious contents, if the feature set is competently chosen. One obvious quandary of this technique is how users pick their function set and the way rich it's. If the characteristic set is very small then most of the malicious content material will not be traced. However, the bigger person function set, better the performance won.

### LIMITATIONS OF EXISTING SYSTEM

* + - * The system is not implemented Learning Algorithms like svm, Naive Bayes.
      * The system is not implemented any the problems involving social networking like privacy, online bullying, misuse, and trolling and many others.

**LITERATURE SURVEY**

With the proliferation of social media platforms, the presence of fake profiles has become a significant concern due to their potential to propagate misinformation, spam, and malicious activities. Detecting these fake profiles efficiently is crucial for maintaining the integrity and trustworthiness of online communities. In recent years, researchers have increasingly turned to machine learning (ML) and natural language processing (NLP) techniques to tackle this problem. This literature survey provides an overview of the existing approaches, methodologies, and advancements in fake profile identification on social networks leveraging ML and NLP techniques. The survey covers key research works, methodologies, datasets, evaluation metrics, and challenges encountered in this domain. Furthermore, it identifies emerging trends and potential future directions for research in combating fake profiles on social media platforms.

1. Introduction

- Background and motivation

- Importance of fake profile identification

- Role of ML and NLP techniques

2. Fake Profile Characteristics

- Behavioral patterns

- Linguistic features

- Network characteristics

3. Existing Approaches

- Rule-based methods

- Supervised learning approaches

- Unsupervised learning techniques

- Hybrid models combining ML and NLP

4. Methodologies

- Feature extraction

- Model architectures

- Training strategies

5. Datasets

- Commonly used datasets for fake profile detection

- Characteristics and limitations

6. Evaluation Metrics

- Performance metrics for assessing fake profile detection models

- Challenges in evaluation

7. Challenges and Limitations

- Data scarcity and imbalance

- Adversarial attacks

- Generalization issues

8. Emerging Trends

- Deep learning approaches

- Cross-platform detection

- Explainable AI techniques

9. Future Directions

- Addressing privacy concerns

- Real-time detection mechanisms

- Incorporating user feedback

10. Conclusion

- Summary of key findings

- Importance of ongoing research in combating fake profiles

11. References

This literature survey aims to provide researchers, practitioners, and policymakers with a comprehensive understanding of the state-of-the-art techniques and challenges in identifying fake profiles on social networks using ML and NLP approaches. By synthesizing existing knowledge and highlighting future directions, this survey seeks to contribute to the advancement of techniques for ensuring the authenticity and trustworthiness of online communities.

In recent years, the surge of fake profiles across social networks has posed significant challenges, ranging from spreading misinformation to facilitating malicious activities. To combat this issue effectively, researchers have turned to machine learning (ML) and natural language processing (NLP) techniques, leveraging the wealth of data available on these platforms. The identification of fake profiles entails discerning subtle behavioral patterns, linguistic anomalies, and network characteristics that deviate from genuine user accounts. Various methodologies have been explored, including rule-based approaches, supervised learning models, unsupervised techniques, and hybrid frameworks combining ML and NLP methodologies. However, despite the advancements, several challenges persist, such as data scarcity, class imbalance, and susceptibility to adversarial attacks. Furthermore, ethical considerations surrounding user privacy and algorithmic biases necessitate careful attention. Nevertheless, ongoing research efforts continue to push the boundaries, with emerging trends focusing on deep learning approaches, cross-platform detection, and explainable AI techniques. Moreover, the industry is witnessing practical applications of fake profile detection systems in social media platforms, online marketplaces, and cybersecurity. In parallel, educational initiatives are crucial in raising awareness among users and integrating digital literacy programs to foster safer online communities. Thus, the interdisciplinary collaboration between computer science, social sciences, and industry stakeholders remains essential in combating the pervasive issue of fake profiles in social networks.

Addressing the ethical implications and ensuring transparency in the deployment of fake profile identification systems is paramount. Moreover, integrating insights from psychology, sociology, and criminology can provide a deeper understanding of fake profile creation and behavior. Industry applications demonstrate the practical relevance of these detection systems in safeguarding online platforms and user communities.

### PROPOSED SYSTEM

On this paper we presented a machine learning & natural language processing system to observe the false profiles in online social networks. Moreover, we are adding the SVM classifier and naïve bayes algorithm to increase the detection accuracy rate of the fake profiles.

An SVM classifies information by means of finding the exceptional hyperplane that separates all information facets of 1 type from those of the other classification. The best hyperplane for an SVM method that the one with the biggest line between the two classes. An SVM classifies data through discovering the exceptional hyperplane that separates all knowledge facets of one category from those of the other class. The help vectors are the info aspects which are closest to the keeping apart hyperplane.

Naive Bayes algorithm is the algorithm that learns the chance of an object with designated features belonging to a unique crew/category. In brief, it's a probabilistic classifier. The Naive Bayes algorithm is called "naive" on account that it makes the belief that the occurrence of a distinct feature is independent of the prevalence of other aspects. For illustration, if we're looking to determine false profiles based on its time, date of publication or posts, language and geoposition. Even if these points depend upon each and every different or on the presence of the other facets, all of these properties in my view contribute to the probability that the false profile.

### ADVANTAGES OF PROPOSED SYSTEM

* + - * In the proposed system, Profile information in online networks will also be static or dynamic. The details which can be supplied with the aid of the person on the time of profile creation is known as static knowledge, the place as the small print that are recounted with the aid of the system within the network is called dynamic knowledge.
      * In the proposed system, Social Networking offerings have facilitated identity theft and Impersonation attacks for serious as good as naïve attackers.

### 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

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

### ECONOMIC FEASIBILITY STUDY

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.

### 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 on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### 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.

### HARDWARE & SOFTWARE REQUIREMENTS

### HARDWARE REQUIREMENTS

* Processor : Pentium –IV
* RAM : 4 GB
* Hard Disk : 20GB
* System : Windows

### SOFTWARE REQUIREMENTS

* Operating system : Windows 7 Ultimate.
* Coding Language : Python.
* Framework : Django-ORM
* Designing : Html, css, javascript.
* Data Base **:** MySQL (WAMP Server)

# ARCHIITECTURE

## ARCHITECTURE

### PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

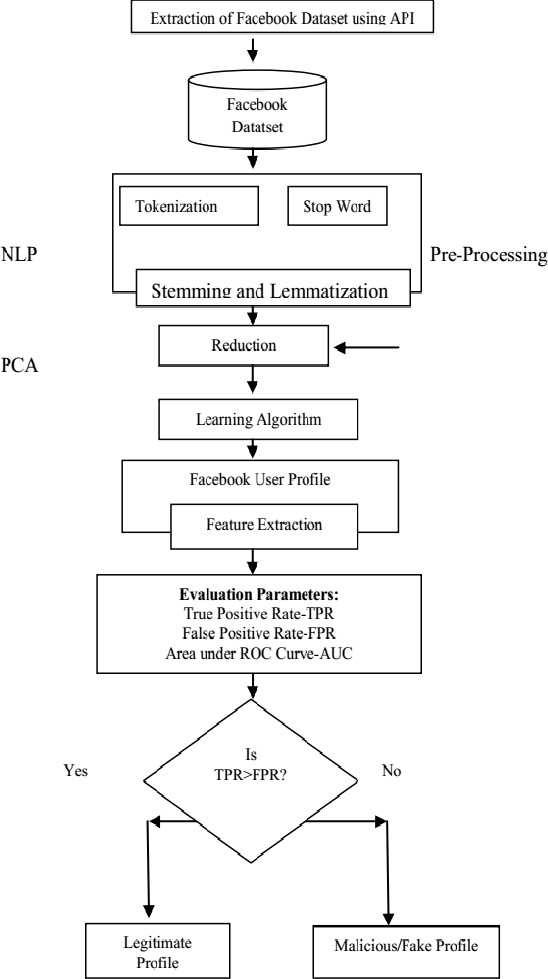


Figure 3.1: Architecture diagram for Fake Profile Identification in Social Network using Machine Learning and NLP.

### DESCRIPTION

### WEB SERVER:

The web server acts as the gateway between the user interface and the backend services. It receives user requests through the web browser, processes these requests, and delivers the corresponding responses. It is responsible for handling static content, managing user sessions, and facilitating communication between the user interface and the underlying services.

### WEB DATABASE:

The web database serves as the central repository for storing and retrieving data critical to the application's functionality. It houses user information, application data, and any other relevant datasets. This relational or non-relational database is integral for maintaining data consistency, integrity, and providing efficient data access to support various features and functionalities of the web application.

### SERVICE PROVIDER

The service provider plays a crucial role in delivering dynamic functionalities and services to the web application. This may include third-party services, APIs, or in-house services that augment the core functionality of the application. These services could range from authentication and authorization services to external integrations, enhancing the overall user experience and expanding the capabilities of the web application.

### REMOTE USERS ACCESS

Remote users access the web application from external locations, such as through the internet, utilizing web browsers or dedicated client applications. This introduces considerations for network security, scalability, and ensuring a responsive user experience despite potential latency.

### 3.3 USECASE DIAGRAM

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

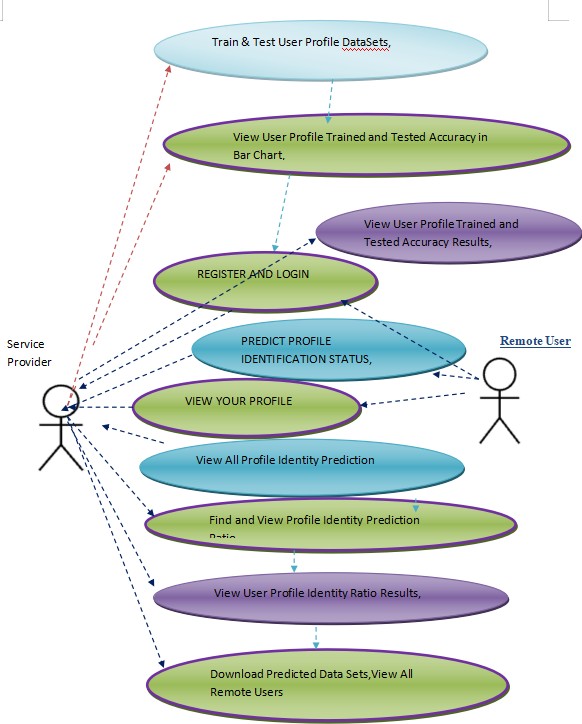


Figure 3.3: UseCase diagram for Fake Profie Identification in Social Network using Machine Learning and NLP.

### 3.4 CLASS DIAGRAM

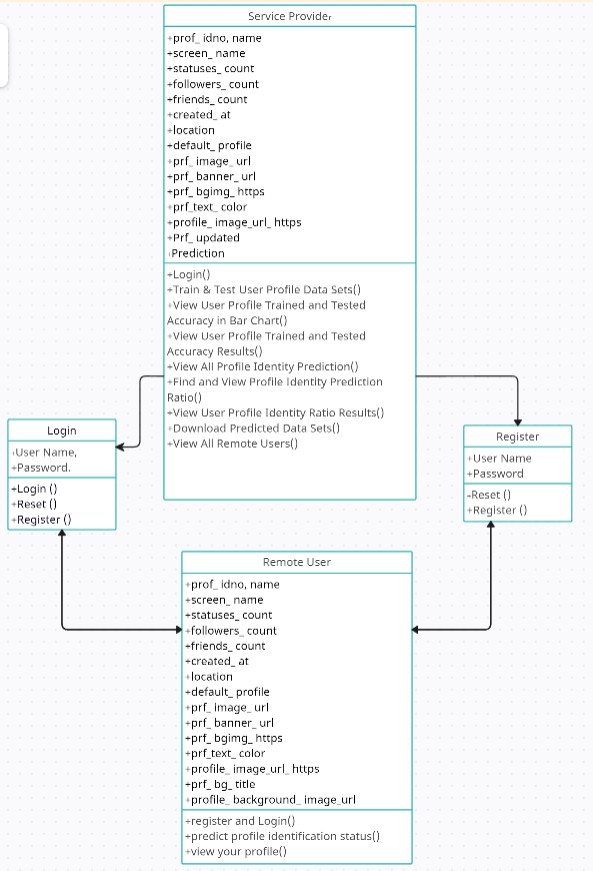
Class diagram is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, operations (or methods), and the relationships among objects.

Figure 3.4: Class diagram for Fake Profie Identification in Social Network using Machine Learning and NLP.

### SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

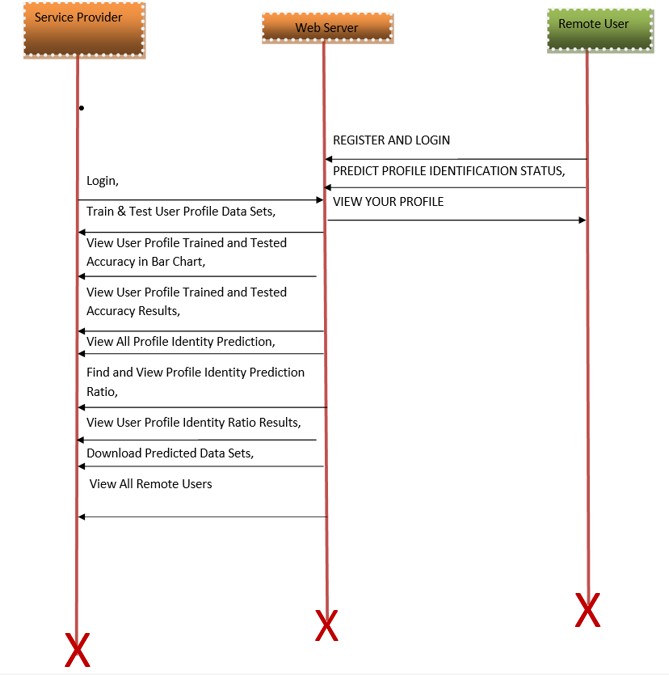
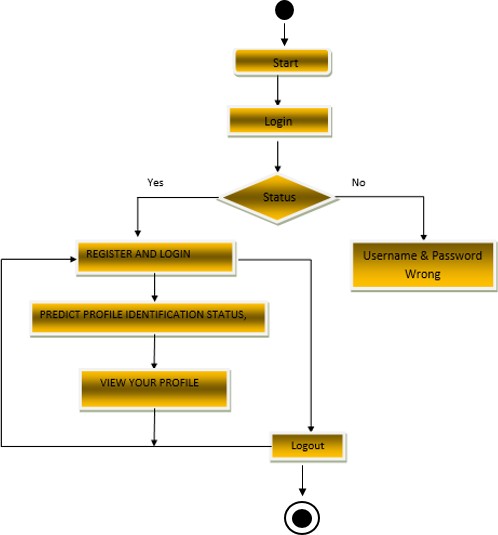


Figure 3.5: Sequence diagram for Fake Profie Identification in Social Network using Machine Learning and NLP.

### ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.

**REMOTE USER :**



**SERVICE PROVIDER:**

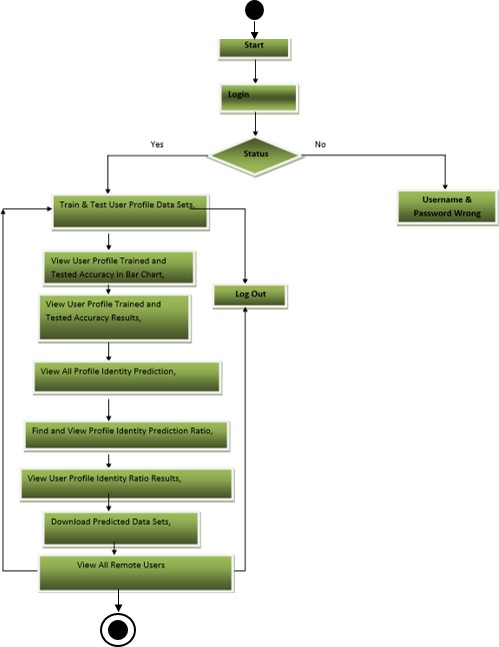


Figure 3.6: Activity diagram for Fake Profie Identification in Social Network using Machine Learning and NLP.

# IMPLEMENTATION

## PYTHON

## Below are some facts about Python.

## • Python is currently the most widely used multi-purpose, high-level programming language.

## • Python allows programming in Object-Oriented and procedural paradigms. Python programs generally are smaller than other programming languages like Java.

## • Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

## • Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

## • The biggest strength of Python is huge collection of standard libraries which can be used for the following –

## o GUI Applications (like Kivy, Tkinter, PyQt etc. )

## o Web frameworks like Django (used by YouTube, Instagram, Dropbox)

## o Image processing (like Open-cv, Pillow)

## o Web scraping (like Scrapy, Beautiful-Soup, Selenium)

## o Test frameworks o Multimedia

## ADVANTAGES OF PYTHON

## Let’s see how Python dominates over other languages.

## Extensive Libraries

## Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don’t have to write the complete code for that manually.

## Extensible

## Python can be extended to other languages. One can write some of the code in languages like C++ or C. This comes in handy, especially in projects.

## Embeddable

## Complimentary to extensibility, Python is embeddable as well. One can put the Python code in the source code of a different language, like C++. This lets by add scripting capabilities to the code in the other language.

## IMPROVED ACCURACY

## The language’s simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that need to write less and get more things done.

## IOT Opportunities

## Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

## Readable

## Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. These further aids the readability of the code.

## Free and Open-Source

## Like it said earlier, Python is freely available. But not only can download Python for free, but one can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help it with one’s tasks.

## Portable

## When code in the project in a language like C++, it may need to make some changes to it if you want to run it on another platform. But it isn’t the same with Python. Here, they need to code only once, and one can run it anywhere. This is called Write Once Run Anywhere (WORA). However, it needs to be careful enough not to include any system-dependent features.

## Interpreted

## Lastly, it will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

## DISADVANTAGES OF PYTHON

## So far, we’ve seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let’s now see the downsides of choosing Python over another language.

## Speed Limitations

## We have seen that Python code is executed line by line. But since python is interpreted, it often results in slow execution. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

## Week in Mobile Computing Devices

## While it serves as an excellent server-side language, Python is much rarely seen on the client- side. Besides that, it is rarely ever used to implement smartphone- based applications. One such application is called Carbonnelle

## Design Restrictions

## As you know, Python is dynamically-typed. This means that you don’t need to declare the type of variable while writing the code. It uses duck-typing. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

## Undeveloped Data Access Layer

## Compared to more widely used technologies like JDBC (Java Data Base Connectivity) and ODBC (Open Data Base Connectivity), Python’s database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

## HISTORY OF PYTHON

## • Python laid its foundation in the late 1980s.

## • In February 1991, Guido Van Rossum published the code (labeled version 0.9.0) to

## • alternate sources.

## • In 1994, Python 1.0 was released with new features like lambda, map, filter, and reduce.

## • Python 2.0 added new features such as list comprehensions, garbage collection systems. Stroke Risk Prediction using Machine Learning algorithms

## • On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify the fundamental flaw of the language.

## • Python 2.0 added new features such as list comprehensions, garbage collection systems. • On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify the fundamental flaw of the language.

## • The implementation of Python was started in December 1989 by Guido Van Rossum at CWI in Netherland.

## • The following programming languages influence Python:

## o ABC language.

## o Modula-3

## MODULES USED

## Pandas

## Pandas is an open-source library in Python. It provides ready to use high- performance data structures and data analysis tools. Pandas’ module runs on top of NumPy and it is popularly used for data science and data analytics. Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data. To install Pandas using pip, enter pip install pandas or pip3 install pandas in the terminal or command line. To install Pandas using condo, execute the following command on a terminal or command line: condo install pandas. Import Error usually arises when the Python working environment doesn't recognize pandas. Pandas has been one of the most commonly used tools for Data Science and Machine learning, which is used for data cleaning and analysis. Here, Pandas is the best tool for handling this real-world messy data. And pandas are one of the open-source python packages built on top of NumPy.

## Seaborn

## Seaborn is a Python data visualization library based on matplotlib. It provides a high- level interface for drawing attractive and informative statistical graphics. In order to install the Seaborn library in Python, you can use either “pip install seaborn” or “condo install seaborn”, depending on which package manager you use.

## Numpy

## Numpy is a general-purpose array-processing package. It provides a high- performance multidimensional array object, and tools for working with these arrays.

## It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

## • A powerful N-dimensional array object.

## • Sophisticated (broadcasting) functions.

## • Tools for integrating C/C++ and Fortran code.

## OpenCV

## OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

## Sklearn

## Scikit-Learn, also known as sklearn is a python library to implement machine learning models and statistical modelling. Through scikit-learn, we can implement various machine learning models for regression, classification, clustering, and statistical tools for analyzing these models. Sklearn model implements a range of machine learning, pre-processing, cross- validation, and visualization algorithms using a unified interface. Important features of scikit- learn: Simple and efficient tools for data mining and data analysis.

## How to Install Scikit-Learn:

## • Download SciPy installer using the link SciPy: Scientific Library for Python - Browse

## • /scipy/0.16. 1 at SourceForge.net.

## • Install Pip bytyping python get\_pip.py in the command line terminal.

## • Install scikit-learn bytyping pip install scikit-learn in the command line.

## Sklearn model

## Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python.

## It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.

## Sklearn metrics

## The sklearn. metrics module implements several loss, score, and utility functions to measure classification performance. Some metrics might require probability estimates of the positive class, confidence values, or binary decisions values.

## Sklearn accuracy

## Sklearn provides a simple function accuracy\_score to compute the Accuracy score. The function takes two arrays as input: y\_true and y\_pred . y\_true is an array of true labels, and y\_pred is an array of predicted labels. The function returns the accuracy score, which is the proportion of correctly classified examples. It takes two arguments: the true labels and the predicted labels. It returns a value between 0 and 1, where 1 represents a perfect prediction and 0 represents a complete mismatch.

## SAMPLE CODE

#!/usr/bin/env python

"""Django's command-line utility for administrative tasks.""" import os

import sys def main():

"""Run administrative tasks.""" os.environ.setdefault('DJANGO\_SETTINGS\_MODULE',

'fake\_profile\_identification.settings') try:

from django.core.management import execute\_from\_command\_line except ImportError as exc:

raise ImportError(

"Couldn't import Django. Are you sure it’s installed and " "available on your PYTHONPATH environment variable? Did you " "forget to activate a virtual environment?"

) from exc execute\_from\_command\_line(sys.argv)

if name == ' main ': main()

from django.db.models import Count, Avg

from django.shortcuts import render, redirect

from django.db.models import Count

from django.db.models import Q

import datetime

import xlwt

from django.http import HttpResponse

import string

import re

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import re

from sklearn.ensemble import VotingClassifier

import warnings

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report

from sklearn.metrics import accuracy\_score

from sklearn.metrics import f1\_score

import numpy as np

import pandas as pd

from sklearn.tree import DecisionTreeClassifier

import openpyxl

# Create your views here.

from Remote\_User.models import

ClientRegister\_Model,profile\_identification\_type,detection\_ratio,detection\_accuracy

def serviceproviderlogin(request):

if request.method == "POST":

admin = request.POST.get('username')

password = request.POST.get('password')

if admin == "Admin" and password =="Admin":

return redirect('View\_Remote\_Users')

return render(request,'SProvider/serviceproviderlogin.html')

def View\_Profile\_Identity\_Prediction(request):

obj = profile\_identification\_type.objects.all()

return render(request, 'SProvider/View\_Profile\_Identity\_Prediction.html', {'objs': obj})

def View\_Profile\_Identity\_Prediction\_Ratio(request):

detection\_ratio.objects.all().delete()

ratio = ""

kword = 'Genuine Profile'

print(kword)

obj = profile\_identification\_type.objects.all().filter(Prediction=kword)

obj1 = profile\_identification\_type.objects.all()

count = obj.count();

count1 = obj1.count();

ratio = (count / count1) \* 100

if ratio != 0:

detection\_ratio.objects.create(names=kword, ratio=ratio)

ratio1 = ""

kword1 = 'Fake Profile'

print(kword1)

obj1 = profile\_identification\_type.objects.all().filter(Prediction=kword1)

obj11 = profile\_identification\_type.objects.all()

count1 = obj1.count();

count11 = obj11.count();

ratio1 = (count1 / count11) \* 100

if ratio1 != 0:

detection\_ratio.objects.create(names=kword1, ratio=ratio1)

obj = detection\_ratio.objects.all()

return render(request, 'SProvider/View\_Profile\_Identity\_Prediction\_Ratio.html', {'objs': obj})

def View\_Remote\_Users(request):

obj=ClientRegister\_Model.objects.all()

return render(request,'SProvider/View\_Remote\_Users.html',{'objects':obj})

def charts(request,chart\_type):

chart1 = detection\_ratio.objects.values('names').annotate(dcount=Avg('ratio'))

return render(request,"SProvider/charts.html", {'form':chart1, 'chart\_type':chart\_type})

def charts1(request,chart\_type):

chart1 = detection\_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))

return render(request,"SProvider/charts1.html", {'form':chart1, 'chart\_type':chart\_type})

def likeschart(request,like\_chart):

charts =detection\_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))

return render(request,"SProvider/likeschart.html", {'form':charts, 'like\_chart':like\_chart})

def likeschart1(request,like\_chart):

charts =detection\_ratio.objects.values('names').annotate(dcount=Avg('ratio'))

return render(request,"SProvider/likeschart1.html", {'form':charts, 'like\_chart':like\_chart})

def Download\_Trained\_DataSets(request):

response = HttpResponse(content\_type='application/ms-excel')

# decide file name

response['Content-Disposition'] = 'attachment; filename="Predicted\_Datasets.xls"'

# creating workbook

wb = xlwt.Workbook(encoding='utf-8')

# adding sheet

ws = wb.add\_sheet("sheet1")

# Sheet header, first row

row\_num = 0

font\_style = xlwt.XFStyle()

# headers are bold

font\_style.font.bold = True

# writer = csv.writer(response)

obj = profile\_identification\_type.objects.all()

data = obj # dummy method to fetch data.

for my\_row in data:

row\_num = row\_num + 1

ws.write(row\_num, 0, my\_row.prof\_idno, font\_style)

ws.write(row\_num, 1, my\_row.name, font\_style)

ws.write(row\_num, 2, my\_row.screen\_name, font\_style)

ws.write(row\_num, 3, my\_row.statuses\_count, font\_style)

ws.write(row\_num, 4, my\_row.followers\_count, font\_style)

ws.write(row\_num, 5, my\_row.friends\_count, font\_style)

ws.write(row\_num, 6, my\_row.created\_at, font\_style)

ws.write(row\_num, 7, my\_row.location, font\_style)

ws.write(row\_num, 8, my\_row.default\_profile, font\_style)

ws.write(row\_num, 9, my\_row.prf\_image\_url, font\_style)

ws.write(row\_num, 10, my\_row.prf\_banner\_url, font\_style)

ws.write(row\_num, 11, my\_row.prf\_bgimg\_https, font\_style)

ws.write(row\_num, 12, my\_row.prf\_text\_color, font\_style)

ws.write(row\_num, 13, my\_row.profile\_image\_url\_https, font\_style)

ws.write(row\_num, 14, my\_row.prf\_bg\_title, font\_style)

ws.write(row\_num, 15, my\_row.profile\_background\_image\_url, font\_style)

ws.write(row\_num, 16, my\_row.description, font\_style)

ws.write(row\_num, 17, my\_row.Prf\_updated, font\_style)

ws.write(row\_num, 18, my\_row.Prediction, font\_style)

wb.save(response)

return response

def Train\_Test\_DataSets(request):

detection\_accuracy.objects.all().delete()

df = pd.read\_csv('Profile\_Datasets.csv')

def clean\_text(text):

'''Make text lowercase, remove text in square brackets,remove links,remove punctuation

and remove words containing numbers.'''

text = text.lower()

text = re.sub('\[.\*?\]', '', text)

text = re.sub('https?://\S+|www\.\S+', '', text)

text = re.sub('<.\*?>+', '', text)

text = re.sub('[%s]' % re.escape(string.punctuation), '', text)

text = re.sub('\n', '', text)

text = re.sub('\w\*\d\w\*', '', text)

text = re.sub('"@', '', text)

text = re.sub('@', '', text)

text = re.sub('https: //', '', text)

text = re.sub('Ã¢â‚¬â€', '', text)

text = re.sub('\n\n', '', text)

return text

df['processed\_content'] = df['name'].apply(lambda x: clean\_text(x))

def apply\_results(label):

if (label == 0):

return 0 # Fake

elif (label == 1):

return 1 # Genuine

df['results'] = df['Label'].apply(apply\_results)

cv = CountVectorizer(lowercase=False)

y = df['results']

X = df["id"].apply(str)

print("X Values")

print(X)

print("Labels")

print(y)

X = cv.fit\_transform(X)

models = []

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

X\_train.shape, X\_test.shape, y\_train.shape

print("X\_test")

print(X\_test)

print(X\_train)

print("Naive Bayes")

from sklearn.naive\_bayes import MultinomialNB

NB = MultinomialNB()

NB.fit(X\_train, y\_train)

predict\_nb = NB.predict(X\_test)

naivebayes = accuracy\_score(y\_test, predict\_nb) \* 100

print("ACCURACY")

print(naivebayes)

print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, predict\_nb))

print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, predict\_nb))

detection\_accuracy.objects.create(names="Naive Bayes", ratio=naivebayes)

# SVM Model

print("SVM")

from sklearn import svm

lin\_clf = svm.LinearSVC()

lin\_clf.fit(X\_train, y\_train)

predict\_svm = lin\_clf.predict(X\_test)

svm\_acc = accuracy\_score(y\_test, predict\_svm) \* 100

print(svm\_acc)

print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, predict\_svm))

print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, predict\_svm))

models.append(('svm', lin\_clf))

detection\_accuracy.objects.create(names="SVM", ratio=svm\_acc)

print("KNeighborsClassifier")

from sklearn.neighbors import KNeighborsClassifier

kn = KNeighborsClassifier()

kn.fit(X\_train, y\_train)

knpredict = kn.predict(X\_test)

print("ACCURACY")

print(accuracy\_score(y\_test, knpredict) \* 100)

print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, knpredict))

print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, knpredict))

models.append(('KNeighborsClassifier', kn))

detection\_accuracy.objects.create(names="KNeighborsClassifier", ratio=accuracy\_score(y\_test, knpredict) \* 100)

obj = detection\_accuracy.objects.all()

return render(request,'SProvider/Train\_Test\_DataSets.html', {'objs': obj})

from django.db.models import Count

from django.db.models import Q

from django.shortcuts import render, redirect, get\_object\_or\_404

import datetime

import openpyxl

import string

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import re

from sklearn.ensemble import VotingClassifier

import warnings

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report

from sklearn.metrics import accuracy\_score

from sklearn.metrics import f1\_score

import numpy as np

import pandas as pd

from sklearn.tree import DecisionTreeClassifier

# Create your views here.

from Remote\_User.models import ClientRegister\_Model,profile\_identification\_type,detection\_ratio,detection\_accuracy

def login(request):

if request.method == "POST" and 'submit1' in request.POST:

username = request.POST.get('username')

password = request.POST.get('password')

try:

enter = ClientRegister\_Model.objects.get(username=username,password=password)

request.session["userid"] = enter.id

return redirect('ViewYourProfile')

except:

pass

return render(request,'RUser/login.html')

def Register1(request):

if request.method == "POST":

username = request.POST.get('username')

email = request.POST.get('email')

password = request.POST.get('password')

phoneno = request.POST.get('phoneno')

country = request.POST.get('country')

state = request.POST.get('state')

city = request.POST.get('city')

ClientRegister\_Model.objects.create(username=username, email=email, password=password, phoneno=phoneno,

country=country, state=state, city=city)

return render(request, 'RUser/Register1.html')

else:

return render(request,'RUser/Register1.html')

def ViewYourProfile(request):

userid = request.session['userid']

obj = ClientRegister\_Model.objects.get(id= userid)

return render(request,'RUser/ViewYourProfile.html',{'object':obj})

def Predict\_Profile\_Identification\_Status(request):

expense = 0

kg\_price=0

if request.method == "POST":

prof\_idno= request.POST.get('prof\_idno')

name= request.POST.get('name')

screen\_name= request.POST.get('screen\_name')

statuses\_count= request.POST.get('statuses\_count')

followers\_count= request.POST.get('followers\_count')

friends\_count= request.POST.get('friends\_count')

created\_at= request.POST.get('created\_at')

location= request.POST.get('location')

default\_profile= request.POST.get('default\_profile')

prf\_image\_url= request.POST.get('prf\_image\_url')

prf\_banner\_url= request.POST.get('prf\_banner\_url')

prf\_bgimg\_https= request.POST.get('prf\_bgimg\_https')

prf\_text\_color= request.POST.get('prf\_text\_color')

profile\_image\_url\_https= request.POST.get('profile\_image\_url\_https')

prf\_bg\_title= request.POST.get('prf\_bg\_title')

profile\_background\_image\_url= request.POST.get('profile\_background\_image\_url')

description= request.POST.get('description')

Prf\_updated = request.POST.get('Prf\_updated')

df = pd.read\_csv('Profile\_Datasets.csv')

def clean\_text(text):

'''Make text lowercase, remove text in square brackets,remove links,remove punctuation

and remove words containing numbers.'''

text = text.lower()

text = re.sub('\[.\*?\]', '', text)

text = re.sub('https?://\S+|www\.\S+', '', text)

text = re.sub('<.\*?>+', '', text)

text = re.sub('[%s]' % re.escape(string.punctuation), '', text)

text = re.sub('\n', '', text)

text = re.sub('\w\*\d\w\*', '', text)

text = re.sub('"@', '', text)

text = re.sub('@', '', text)

text = re.sub('https: //', '', text)

text = re.sub('Ã¢â‚¬â€', '', text)

text = re.sub('\n\n', '', text)

return text

df['processed\_content'] = df['name'].apply(lambda x: clean\_text(x))

def apply\_results(label):

if (label == 0):

return 0 # Fake

elif (label == 1):

return 1 # Genuine

df['results'] = df['Label'].apply(apply\_results)

cv = CountVectorizer(lowercase=False)

y = df['results']

X = df["id"].apply(str)

print("X Values")

print(X)

print("Labels")

print(y)

X = cv.fit\_transform(X)

models = []

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

X\_train.shape, X\_test.shape, y\_train.shape

# SVM Model

print("SVM")

from sklearn import svm

lin\_clf = svm.LinearSVC()

lin\_clf.fit(X\_train, y\_train)

predict\_svm = lin\_clf.predict(X\_test)

svm\_acc = accuracy\_score(y\_test, predict\_svm) \* 100

print(svm\_acc)

print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, predict\_svm))

print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, predict\_svm))

models.append(('svm', lin\_clf))

print("KNeighborsClassifier")

from sklearn.neighbors import KNeighborsClassifier

kn = KNeighborsClassifier()

kn.fit(X\_train, y\_train)

knpredict = kn.predict(X\_test)

print("ACCURACY")

print(accuracy\_score(y\_test, knpredict) \* 100)

print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, knpredict))

print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, knpredict))

models.append(('KNeighborsClassifier', kn))

classifier = VotingClassifier(models)

classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test)

prof\_idno1 = [prof\_idno]

vector1 = cv.transform(prof\_idno1).toarray()

predict\_text = classifier.predict(vector1)

pred = str(predict\_text).replace("[", "")

pred1 = pred.replace("]", "")

prediction = int(pred1)

if prediction == 0:

val = 'Fake Profile'

elif prediction == 1:

val = 'Genuine Profile'

print(val)

print(pred1)

profile\_identification\_type.objects.create(

prof\_idno=prof\_idno,

name=name,

screen\_name=screen\_name,

statuses\_count=statuses\_count,

followers\_count=followers\_count,

friends\_count=friends\_count,

created\_at=created\_at,

location=location,

default\_profile=default\_profile,

prf\_image\_url=prf\_image\_url,

prf\_banner\_url=prf\_banner\_url,

prf\_bgimg\_https=prf\_bgimg\_https,

prf\_text\_color=prf\_text\_color,

profile\_image\_url\_https=profile\_image\_url\_https,

prf\_bg\_title=prf\_bg\_title,

profile\_background\_image\_url=profile\_background\_image\_url,

description=description,

Prf\_updated=Prf\_updated,

Prediction=val)

return render(request, 'RUser/Predict\_Profile\_Identification\_Status.html',{'objs':val})

return render(request, 'RUser/Predict\_Profile\_Identification\_Status.html')

# 5.RESULTS

## 5.RESULTS

### 5.1 LOGIN :

User must login with username and password if user doesn’t exist user need to register in this application by necessary details.



FIGURE 5.1: Login

### PREDICTION

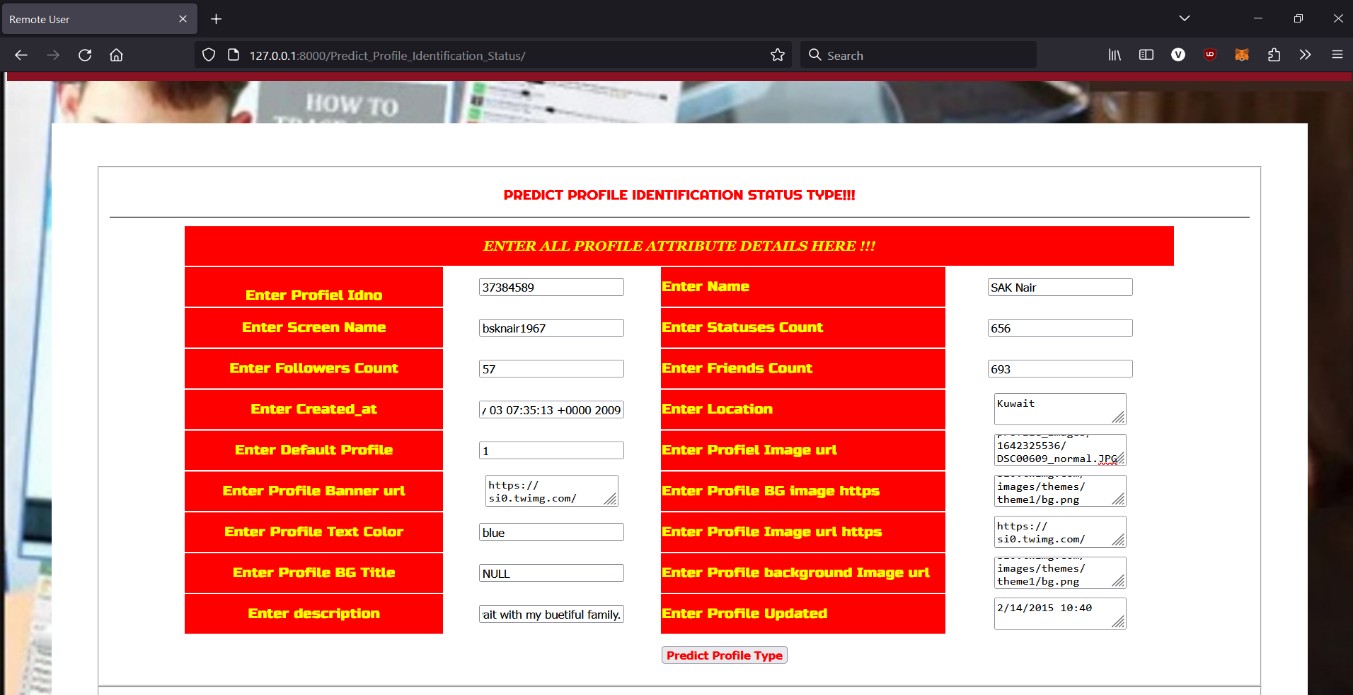
User has to give the profile details to Identify the Fake Profiles in Social Network.

FIGURE 5.2: Status

### IDENTIFYING FAKE PROFILES

After user entering all the profile details user need to click the predict and it will predict whether it is fake profile or genuine profile.

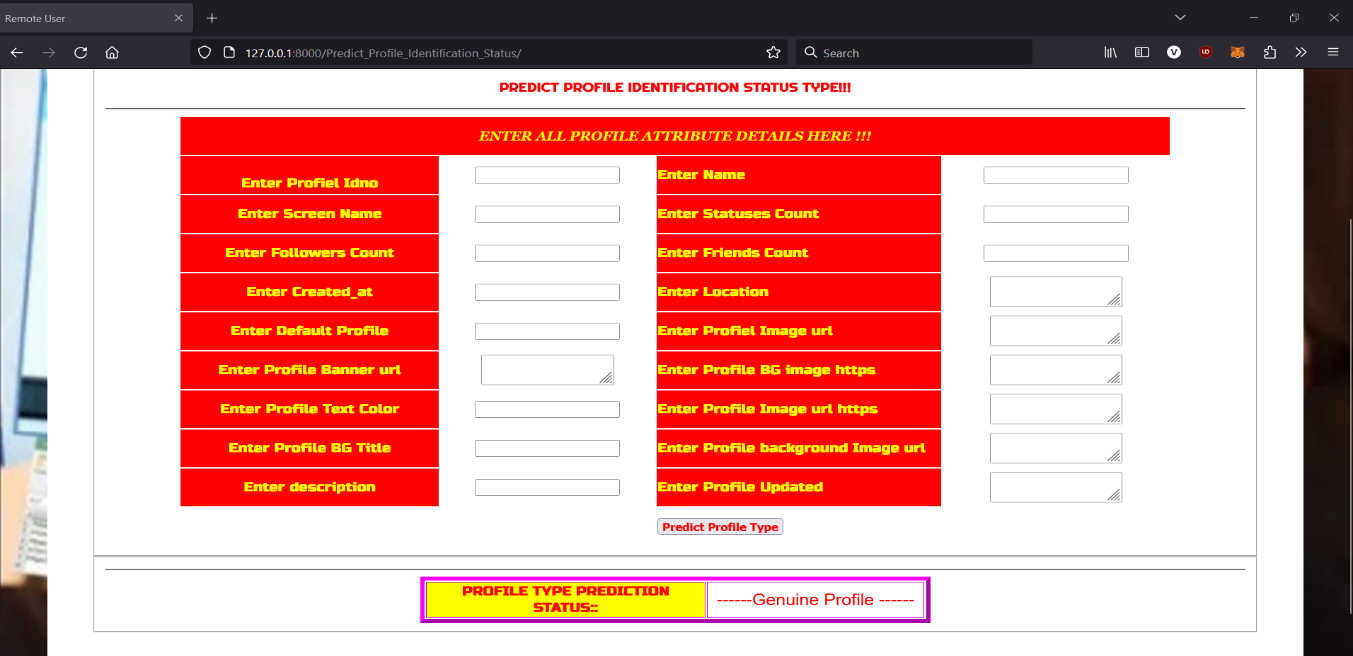


FIGURE 5.3: Profile Type Prediction

### BAR CHART

It shows User profile Trained and Tested Accuracy in Bar Chart.

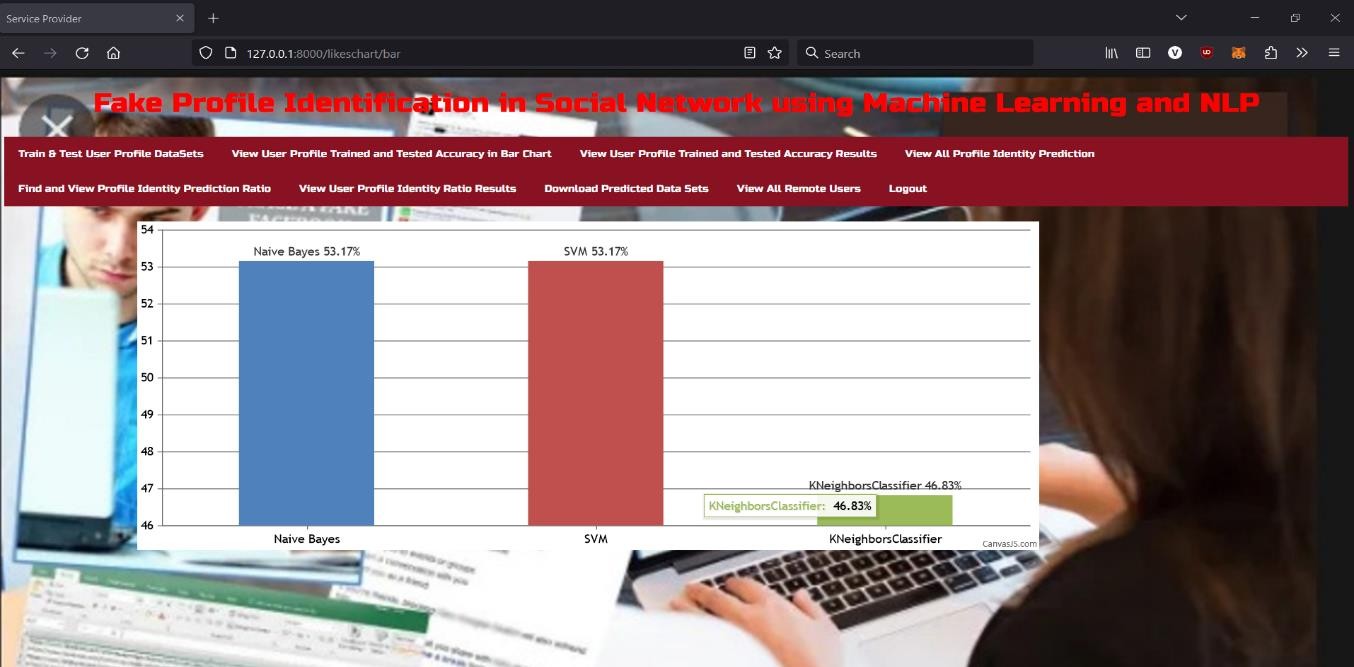


FIGURE 5.4: Bar Chart

# 6.TESTING

## 6. TESTING

### INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement

### 6.1TYPES OF TESTING

* + 1. **UNIT TESTING**

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. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit testsensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

### INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input mustbe rejected. Functions : identified functions must be exercised.

Output : identified classes of application outputsmust be exercised. Systems/Procedures : interfacing systems or procedures must be invoked.

### TEST CASES

* + 1. **TABLE OF CLASSIFICATION**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Expected Outcome** | **Status** |
| 01 | Data Validation Test | Validate that incoming data falls within acceptable ranges. | Passed |
| 02 | Predictive Model Accuracy Test | Validate the accuracy of Fake Profile predictions against actual data. | Passed |
| 03 | User Access Control Test | Check that different user roles have appropriate data access. | Passed |

# 7.CONCLUSION

## 7.CONCLUSION & FUTURE SCOPE

### PROJECT CONCLUSION

In conclusion, employing machine learning (ML) and natural language processing (NLP) techniques for fake profile identification in social networks holds immense promise. By integrating ML algorithms like SVM and Naïve Bayes with NLP methods, we enhance our ability to detect fraudulent profiles with greater accuracy and efficiency. This approach not only improves the overall security and trustworthiness of online platforms but also provides users with a safer and more authentic social networking experience. As we continue to refine and develop these technologies, we move closer to creating a digital environment where users can interact confidently and securely.

### 7.2 FUTURE SCOPE

**Collaboration with Cybersecurity Frameworks**:

Collaboration with cybersecurity frameworks will lead to the development of comprehensive solutions that combine fake profile detection with broader security measures, safeguarding users from various online threats.

**Adaptation to Emerging Platforms**:

As new social media platforms and communication mediums emerge, future efforts will focus on adapting fake profile detection techniques to these platforms, ensuring proactive detection and prevention of fraudulent activities across diverse online environments.

# REFERENCES

### 8.REFERENCES

### 8.1 REFERENCES

1. Michael Fire et al. (2012). "Strangers intrusion detection-detecting spammers and fake profiles in social networks based on topology anomalies." Human Journal 1(1): 26- 39.Günther, F. and S. Fritsch (2010). "neuralnet: Training of neural networks." The R Journal 2(1): 30-38

.

1. Dr. S. Kannan, Vairaprakash Gurusamy, “Preprocessing Techniques for Text Mining”, 05 March 2015.
2. Shalinda Adikari and Kaushik Dutta, Identifying Fake Profiles in LinkedIn, PACIS 2014 Proceedings, AISeL
3. Z. Halim, M. Gul, N. ul Hassan, R. Baig, S. Rehman, and F. Naz,“Malicious users’ circle detection in social network based on spatiotemporal co-occurrence,” Computer Networks and Information Technology (ICCNIT),2011 International Conference on, July, pp. 35–390.
4. Liu Y, Gummadi K, Krishnamurthy B, Mislove A,” Analyzing Facebook privacy settings: User expectations vs. reality” , in: Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference,ACM,pp.61–70.
5. Mahmood S, Desmedt Y,” Poster: preliminary analysis of google?’s privacy. In: Proceedings of the 18th ACM conference on computer and communications security”, ACM 2011, pp.809–812.
6. Stein T, Chen E, Mangla K,” Facebook immune system. In: Proceedings of the 4th workshop on social network systems”, ACM 2011, pp
7. Saeed Abu-Nimeh, T. M. Chen, and O. Alzubi, “Malicious and Spam Posts in Online Social Networks,” Computer, vol.44, no.9, IEEE2011, pp.23 28.
8. J. Jiang, C. Wilson, X. Wang, P. Huang, W. Sha, Y. Dai, B. Zhao, Understanding latent interactions in online social networks, in: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, ACM, 2010, pp. 369–382
9. Kazienko, P. and K. Musiał (2006). Social capital in online social networks. Knowledge-Based Intelligent Information and Engineering Systems, Springer

### GITHUB LINK

https://github.com/nikithareddyk28/indiansignlanguage