

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
V GOPI SINGH (217R5A0523)
THIPPAMOLA MAHESH (207R1A05P3)
PATHURI SHRUTHI REDDY (207R1A05M7)

UNDER THE GUIDANCE OF
K. RANJITH REDDY
(Assistant Professor)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CMR TECHNICAL CAMPUS
UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by
AICTE, New Delhi) 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 **V GOPI SINGH (217R5A0523), THIPPAMOLA MAHESH (207R1A05P3) & PATHURI SHRUTHI REDDY (207R1A05M7)** 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 project have not been submitted to any other University or Institute for the award of any degree or diploma.

K. Ranjith Reddy
(Assistant Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

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. RANJITH REDDY**, Assistant 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 Review Committee (PRC) **G.Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** 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 course of 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.

V GOPI SINGH	(217R5A0523)
THIPPAMOLA MAHESH	(207R1A05P3)
PATHURI SHRUTHI REDDY	(207R1A05M7)

ABSTRACT

At present social network sites are part of the life for most of the people. Every day several people are creating their profiles on the social network platforms and they are interacting with others independent of the user's location and time. The social network sites not only providing advantages to the users and also provide security issues to the users as well their information. To analyze, who are encouraging threats in social network we need to classify the social networks profiles of the users. From the classification, we can get the genuine profiles and fake profiles on the social networks. Traditionally, we have different classification methods for detecting the fake profiles on the social networks. But, we need to improve the accuracy rate of the fakeprofile detection in the social networks. In this project we are proposing Machine learning and Natural language Processing (NLP) techniques to improve the accuracy rate of the fake profiles detection. We can use the Support Vector Machine (SVM) and Naïve Bayes algorithm.

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO
Figure 3.1	Project Architecture of Fake Profile Identification in Social Network using Machine Learning and NLP.	9
Figure 3.2	Use Case Diagram for Fake Profile Identification in Social Network using Machine Learning and NLP.	11
Figure 3.3	Class Diagram for Fake Profile Identification in Social Network using Machine Learning and NLP.	12
Figure 3.4	Sequence diagram for Fake Profile Identification in Social Network using Machine Learning and NLP.	13
Figure 3.5	Data flow diagram for Fake Profile Identification in Social Network using Machine Learning and NLP.	14

LIST OF SCREENSHOTS

SCREENSHOT NO.	SCREENSHOT NAME	PAGE NO.
Screenshot 5.1	Remote User Interface	29
Screenshot 5.2	Register page for remote user	30
Screenshot 5.3	User Profile Details After Registration	31
Screenshot 5.4	Predict Profile Whether Genuine Or Fake.	32
Screenshot 5.5	Service Provider login Interface	33
Screenshot 5.6	All Remote Users Details Who Registered	34
Screenshot 5.7	User Profile Trained And Test Accuracy In Bar Char	35
Screenshot 5.8	User Profile Trained And Test Accuracy In Line Chart	36
Screenshot 5.9	User Profile Trained And Test Accuracy In Pie Chart.	37
Screenshot 5.10	All Profiles Status That Are Predicted.	38
Screenshot 5.11	Profile Status Prediction Type Ratio.	39
Screenshot 5.12	Profile Status Prediction Type Ratio In Line Chart.	40
Screenshot 5.13	Profile Status Prediction Type Ratio In Pie Chart.	41

TABLE OF CONTENTS

ABSTRACT	i
LIST OF FIGURES	ii
LIST OF SCREENSHOTS	iii
1. INTRODUCTION	1
1.1 PROJECT SCOPE	1
1.2 PROJECT PURPOSE	1
1.3 PROJECT FEATURES	2
2. SYSTEM ANALYSIS	3
2.1 PROBLEM DEFINITION	4
2.2 EXISTING SYSTEM	4
2.2.1 DISADVANTAGES OF THE EXISTING SYSTEM	5
2.3 PROPOSED SYSTEM	5
2.3.1 ADVANTAGES OF PROPOSED SYSTEM	5
2.4 FEASIBILITY STUDY	6
2.4.1 ECONOMIC FEASIBILITY	6
2.4.2 TECHNICAL FEASIBILITY	6
2.4.3 SOCIAL FEASIBILITY	7
2.5 HARDWARE & SOFTWARE REQUIREMENTS	7
2.5.1 HARDWARE REQUIREMENTS	7
2.5.2 SOFTWARE REQUIREMENTS	8
3. ARCHITECTURE	9
3.1 PROJECT ARCHITECTURE	9
3.2 DESCRIPTION	10
3.3 USE CASE DIAGRAM	11
3.4 CLASS DIAGRAM	12
3.5 SEQUENCE DIAGRAM	13
3.6 DATA FLOW DIAGRAM	14
4. IMPLEMENTATION	15
4.1 SAMPLE CODE	15
5. SCREENSHOTS	29

6 . TESTING	42
6.1 INTRODUCTION TO TESTING	42
6.2 TYPES OF TESTING	42
6.2.1 UNIT TESTING	42
6.2.2 INTEGRATION TESTING	42
6.2.3 FUNCTIONAL TESTING	43
6.3 TEST CASES	45
7. CONCLUSION & FUTURE SCOPE	46
6.3 PROJECT CONCLUSION	46
6.4 FUTURE SCOPE	46
8. BIBLIOGRAPHY	47
8.1 REFERENCES	47
8.2 GITHUB LINK	47

1.INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

This project focuses on developing a Fake Profile Identification system for social networks by leveraging Machine Learning (ML) and Natural Language Processing (NLP). The scope encompasses the collection and preprocessing of a diverse dataset, feature engineering, and the integration of ML algorithms capable of discerning between genuine and fake profiles. NLP techniques will be employed to analyze textual data, improving the system's ability to identify linguistic patterns indicative of fraudulent accounts. The project aims to deploy a user-friendly solution with continuous monitoring capabilities, emphasizing ethical considerations, user education, and documentation for transparency and long-term sustainability.

1.2 PROJECT PURPOSE

The purpose of the Fake Profile Identification project in Social Networks is to employ advanced Machine Learning (ML) and Natural Language Processing (NLP) techniques to automatically detect and classify fraudulent profiles within social platforms. By analyzing diverse features such as user behavior, network interactions, and linguistic patterns in textual content, the project aims to develop a robust system that can effectively differentiate between genuine and fake profiles. The primary objective is to enhance the security and trustworthiness of social networks by providing users and platform administrators with a reliable tool for identifying and mitigating the presence of deceptive profiles, thereby fostering a safer and more authentic online community.

1.3 PROJECT FEATURES

The Fake Profile Identification project in Social Networks incorporates several key features to achieve its objectives. First, it involves the collection of a diverse dataset comprising both genuine and fake profiles, ensuring a representative sample. The project employs advanced data preprocessing techniques to clean and structure the data appropriately. Feature engineering is utilized to extract relevant information, including profile completeness, posting behavior, and linguistic patterns from textual content.

The integration of Machine Learning (ML) algorithms and Natural Language Processing (NLP) techniques enhances the system's ability to analyze user profiles comprehensively. ML models are trained on the dataset to classify profiles as genuine or fake, and NLP contributes to the analysis of textual information associated with profiles, thereby capturing linguistic nuances indicative of fraudulent activity.

The deployment phase involves creating a user-friendly system with real-time or batch processing capabilities for continuous monitoring of social network profiles. Ethical considerations, user education, and documentation are integrated into the project, ensuring responsible use and transparency. This comprehensive set of features aims to deliver an effective and scalable solution for the automatic identification of fake profiles, contributing to a safer and more secure online social environment.

2. SYSTEM ANALYSIS

2. 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.

2.1 PROBLEM DEFINITION

The Fake Profile Identification project in Social Networks addresses the escalating issue of deceptive profiles within online communities. With the proliferation of fake accounts engaging in malicious activities such as spreading misinformation, scams, and impersonation, there is a critical need for an automated system to distinguish between genuine and fraudulent profiles. Leveraging the power of Machine Learning (ML) and Natural Language Processing (NLP), the project aims to develop a solution that can analyze user behavior, network interactions, and linguistic patterns in textual content to accurately identify and flag suspicious profiles. By mitigating the presence of fake accounts, the project seeks to enhance the security, authenticity, and trustworthiness of social networks, creating a safer digital space for users to interact.

2.2 EXISTING SYSTEM

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. Creation of a false profile turns into such adversary outcomes which is intricate to identify without apt research. The current solutions which were virtually developed and theorized to resolve this contention, mainly viewed the traits and the social network ties of the person's social profile. 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. The limited publicly available profile data of LinkedIn makes it ineligible in making use of the existing tactics in fake profile identification. 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. 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.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

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

2.3 PROPOSED SYSTEM

In this project, we proposed 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.

2.3.1 ADVANTAGES OF THE 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.

2.4 FEASIBILITY STUDY

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

Three key considerations involved in the feasibility analysis are,

- ◆ **ECONOMICAL FEASIBILITY**
- ◆ **TECHNICAL FEASIBILITY**
- ◆ **SOCIAL FEASIBILITY**

2.4.1 ECONOMIC FEASIBILITY

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

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands 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.

2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. 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.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

For developing the application the following are the Hardware Requirement

- **Processor** – Pentium IV or Intel Core i5 or i7, AMD Ryzen 5 or 7, or equivalent processors
- **RAM** -- 256 MB
- **Space on Hard Disk** -- 20 GB

2.5.2 SOFTWARE REQUIREMENTS:

For developing the Application the following are the software Requirement

1. Operating System :

- Windows 7 or Higher

2. Programming language :

- Python

3. Frame Work :

- Django

4. Database :

- MySQL

5. Server :

- Wamp 2

3.ARCHITECTURE

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

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

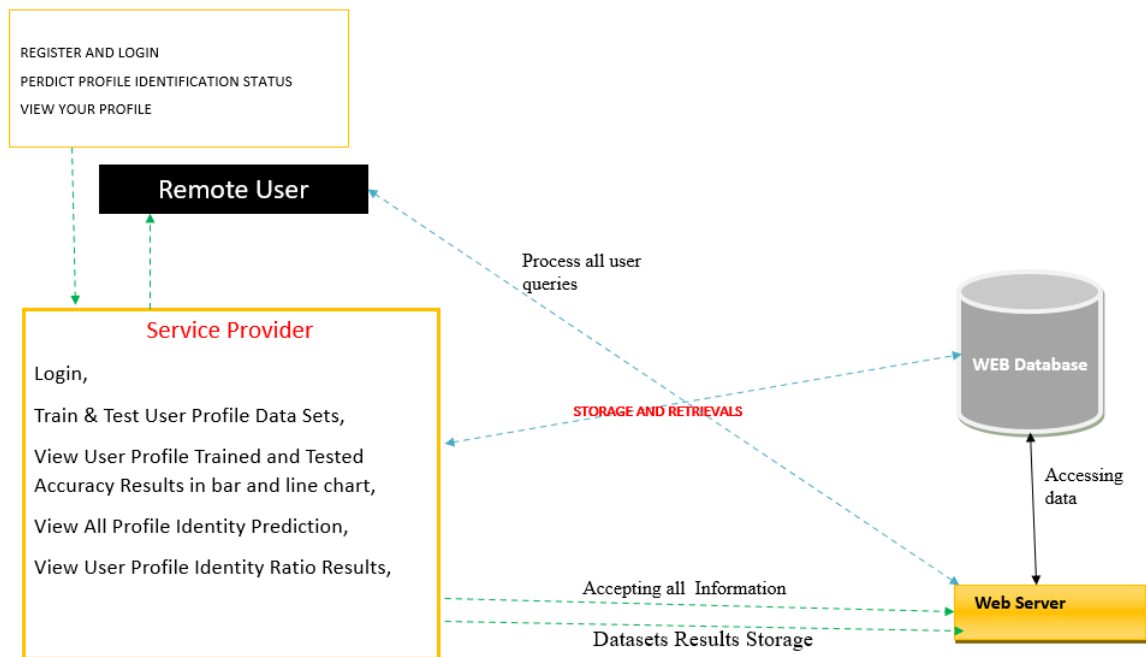


Figure 3.1: Architecture of Fake Profile Identification in Social Network using Machine Learning and NLP

3.2 DESCRIPTION

The architecture for Fake Profile Identification in Social Networks using Machine Learning (ML) and Natural Language Processing (NLP) is designed as a seamless integration of three key phases. Initially, a diverse dataset is collected and preprocessed to extract pertinent features, including profile completeness and linguistic patterns. Subsequently, ML models are integrated to learn from this data, discerning between genuine and fake profiles. Simultaneously, NLP techniques are employed to analyze textual information, enhancing the system's linguistic understanding. The final deployment phase involves implementing a user-friendly interface for real-time profile analysis and continuous monitoring. Ethical considerations and feedback mechanisms are embedded to ensure responsible use and ongoing improvement, resulting in a comprehensive architecture for robust fake profile identification in social networks.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model. 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.

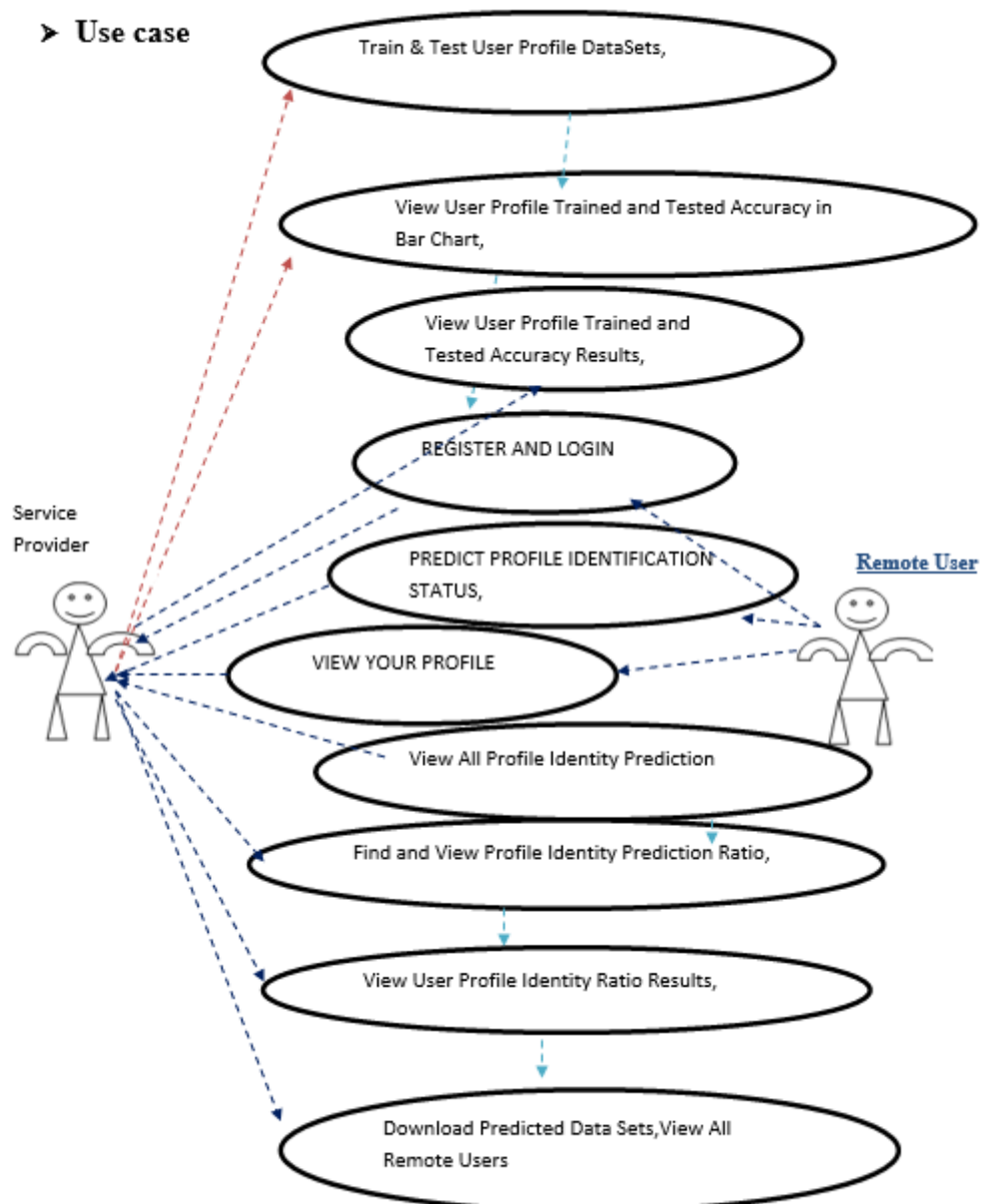


Figure 3.2: Use Case Diagram for Fake Profile Identification in Social Network using Machine Learning and NLP

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.

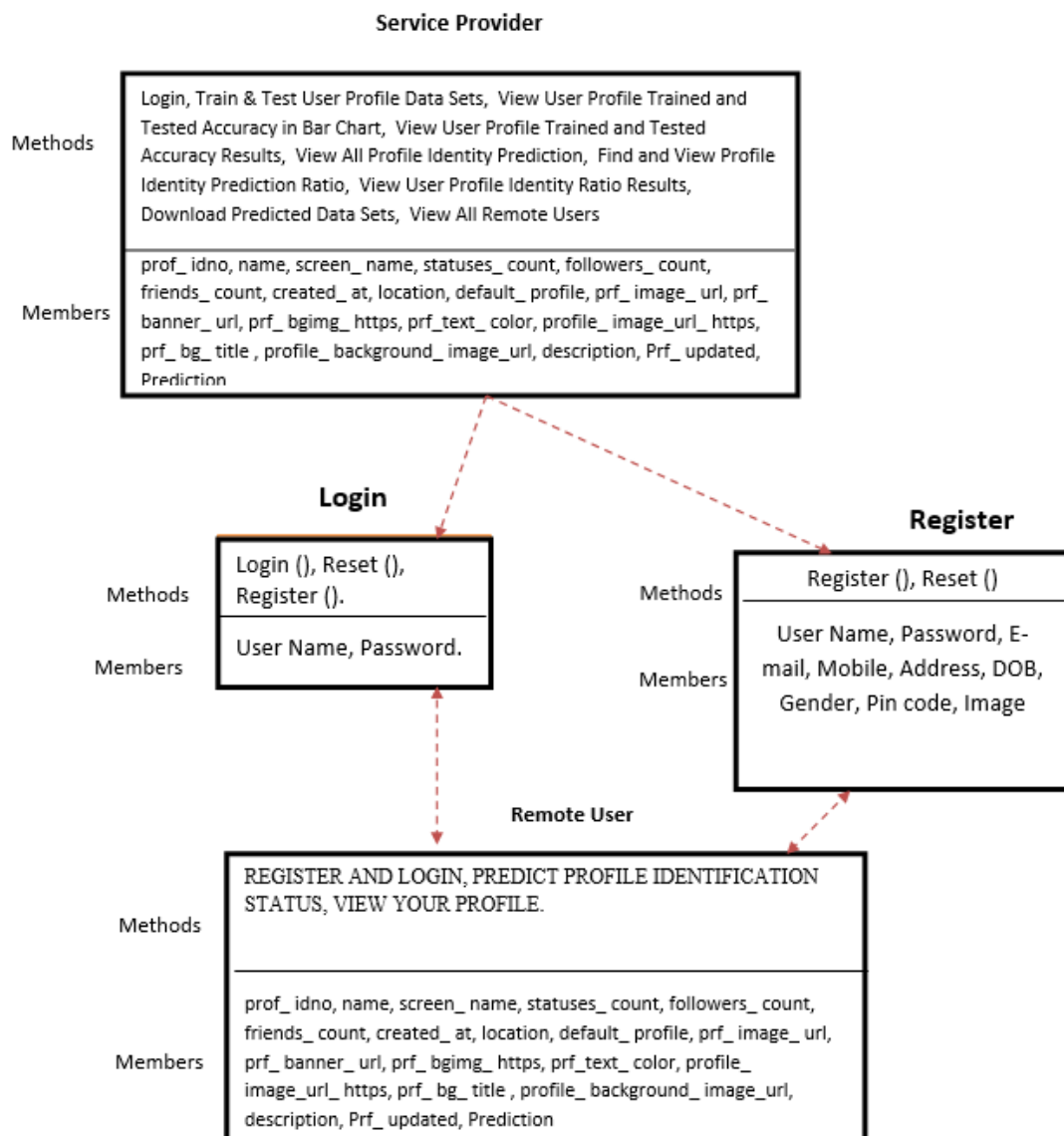


Figure 3.3: Class Diagram for Fake Profile Identification in Social Network using Machine Learning and NLP

3.5 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.

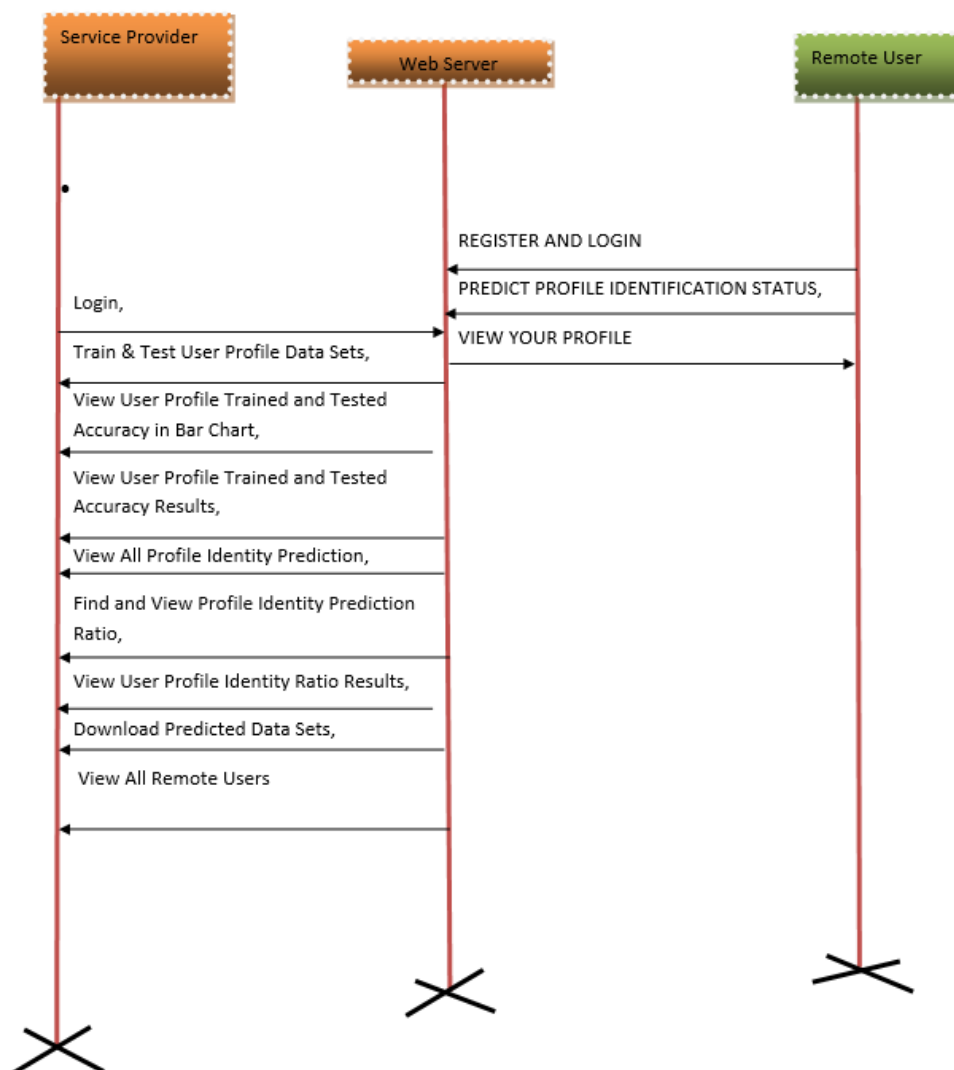


Figure 3.4: Sequence Diagrams for Fake Profile Identification in Social Network using Machine Learning and NLP

3.6 DATA FLOW DIAGRAM

Data flow diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

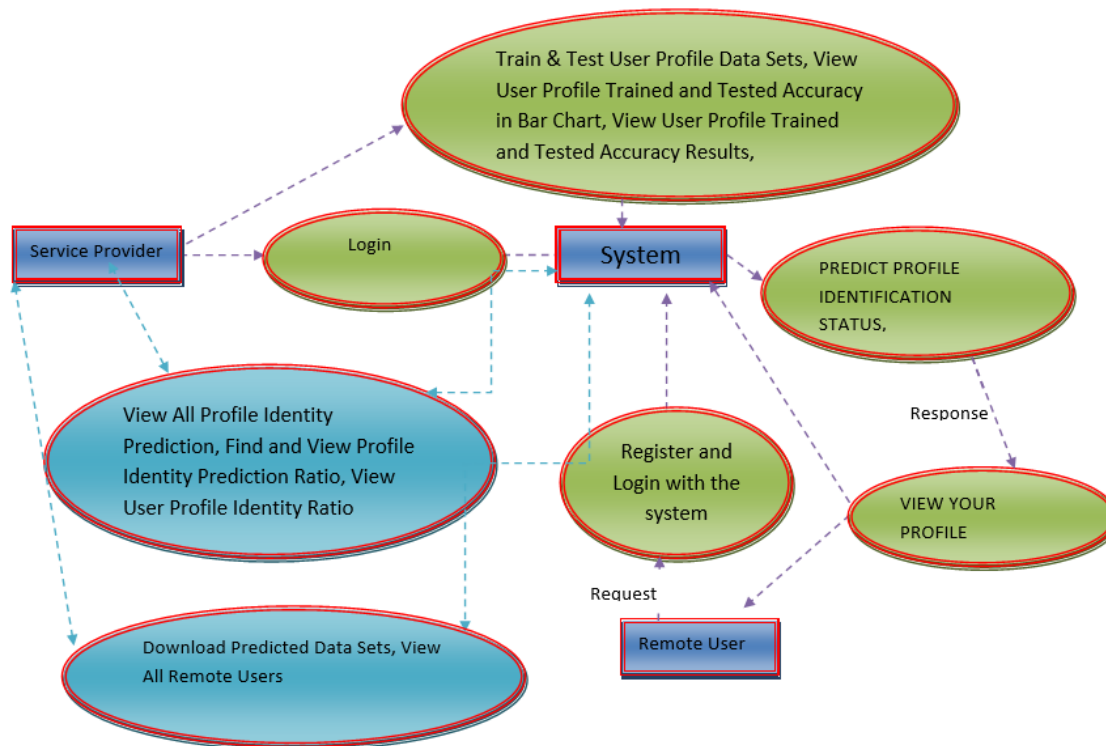
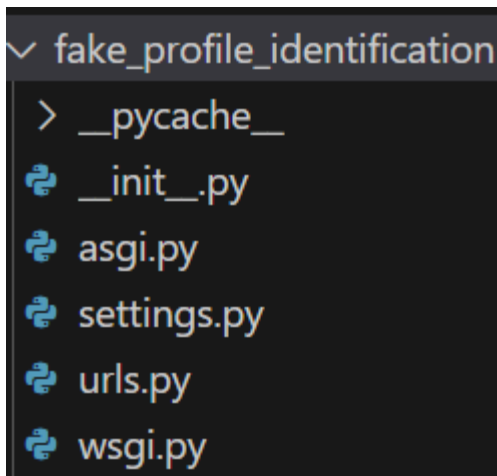


Figure 3.5: Data Flow Diagram for Fake Profile Identification in Social Network using Machine Learning and NLP

4. IMPLEMENTATION

4.1 SAMPLE CODES



asgi.py

```
"""
```

ASGI config for fake_profile_identification

It exposes the ASGI callable as a module-level variable named ``application``.

For more information on this file, see

<https://docs.djangoproject.com/en/3.0/howto/deployment/asgi/>

```
"""
```

```
import os
```

```
from django.core.asgi import get_asgi_application
```

```
os.environ.setdefault('DJANGO_SETTINGS_MODULE',
    'fake_profile_identification.settings')
```

```
application = get_asgi_application()
```

settings.py

```

import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'm+1edl5m-5@u9u!b8-=4-4mq&o1%agco2xpl8c!7sn7!eowjk#'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'Remote_User',
    'Service_Provider',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

```

```

ROOT_URLCONF = 'fake_profile_identification.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [(os.path.join(BASE_DIR, 'Template/htmls'))],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'fake_profile_identification.wsgi.application'

# Database
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'fake_profile_identification',
        'USER': 'root',
        'PASSWORD': '',
        'HOST': '127.0.0.1',
        'PORT': '3306',
    }
}

# Password validation
# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
        'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },

```

```

    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

```

Internationalization

<https://docs.djangoproject.com/en/3.0/topics/i18n/>

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True

Static files (CSS, JavaScript, Images)

<https://docs.djangoproject.com/en/3.0/howto/static-files/>

STATIC_URL = '/static/'

STATICFILES_DIRS = [os.path.join(BASE_DIR, 'Template/images')]

MEDIA_URL = '/media/'

MEDIA_ROOT = os.path.join(BASE_DIR, 'Template/media')

STATIC_ROOT = '/static/'

STATIC_URL = '/static/'

ulrs.py

```
"""fake_profile_identification URL Configuration
```

The `urlpatterns` list routes URLs to views. For more information please see:

<https://docs.djangoproject.com/en/3.0/topics/http/urls/>

Examples:

Function views

1. Add an import: from my_app import views
2. Add a URL to urlpatterns: path("", views.home, name='home')

Class-based views

1. Add an import: from other_app.views import Home
2. Add a URL to urlpatterns: path("", Home.as_view(), name='home')

Including another URLconf

1. Import the include() function: from django.urls import include, path
2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))

```
"""
```

```
from django.conf.urls import url
from django.contrib import admin
from Remote_User import views as remoteuser
from fake_profile_identification import settings
from Service_Provider import views as serviceprovider
from django.conf.urls.static import static
```

```
urlpatterns = [
    url('admin/', admin.site.urls),
    url(r'^$', remoteuser.login, name="login"),
    url(r'^Register1/$', remoteuser.Register1, name="Register1"),
    url(r'^Predict_Profile_Identification_Status/$',
remoteuser.Predict_Profile_Identification_Status,
name="Predict_Profile_Identification_Status"),
    url(r'^ViewYourProfile/$', remoteuser.ViewYourProfile,
name="ViewYourProfile"),
    url(r'^serviceproviderlogin/$',serviceprovider.serviceproviderlogin,
name="serviceproviderlogin"),

url(r'^View_Remote_Users/$',serviceprovider.View_Remote_Users,name="View_Re
mote_Users"),
    url(r'^charts/(?P<chart_type>\w+)', serviceprovider.charts,name="charts"),
    url(r'^charts1/(?P<chart_type>\w+)', serviceprovider.charts1, name="charts1"),
    url(r'^likeschart/(?P<like_chart>\w+)', serviceprovider.likeschart,
name="likeschart"),
```

```

url(r'^View_Profile_Identity_Prediction_Ratio/$',
serviceprovider.View_Profile_Identity_Prediction_Ratio,name="View_Profile_Identi
ty_Prediction_Ratio"),
url(r'^likeschart1/(?P<like_chart1>\w+)', serviceprovider.likeschart1,
name="likeschart1"),
url(r'^Train_Test_DataSets/$', serviceprovider.Train_Test_DataSets,
name="Train_Test_DataSets"),
url(r'^View_Profile_Identity_Prediction/$',
serviceprovider.View_Profile_Identity_Prediction,
name="View_Profile_Identity_Prediction"),
url(r'^Download_Trained_DataSets/$',
serviceprovider.Download_Trained_DataSets,
name="Download_Trained_DataSets"),

]+ static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

wsgi.py

```

"""

```

WSGI config for fake_profile_identification.

It exposes the WSGI callable as a module-level variable named ``application``.

For more information on this file, see

<https://docs.djangoproject.com/en/3.0/howto/deployment/wsgi/>

```

"""

```

```

import os

```

```

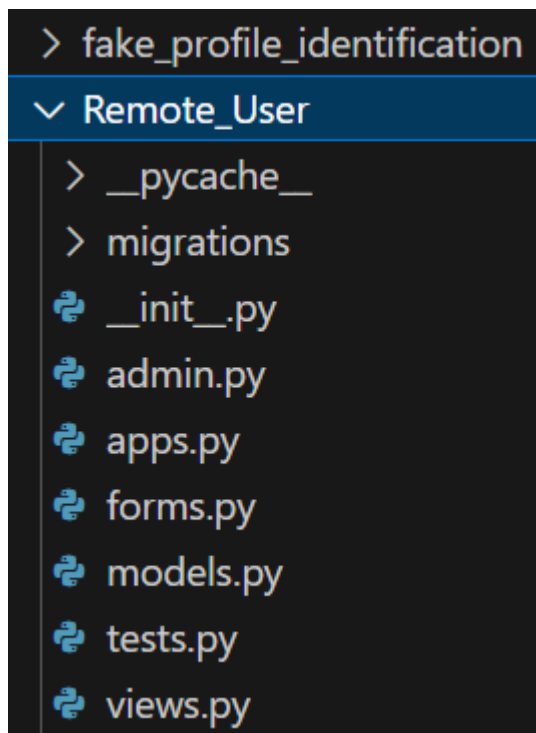
from django.core.wsgi import get_wsgi_application

```

```

os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'fake_profile_identification.settings')
application = get_wsgi_application()

```

Apps.py

```
from django.apps import AppConfig
```

```
class ClientSiteConfig(AppConfig):
    name = 'Remote_User'
```

forms.py

```
from django import forms
```

```
from Remote_User.models import ClientRegister_Model
```

```
class ClientRegister_Form(forms.ModelForm):
    password = forms.CharField(widget=forms.PasswordInput())
    email = forms.EmailField(required=True)
```

```
class Meta:
    model = ClientRegister_Model
    fields = ("username", "email", "password", "phoneno", "country", "state", "city")
```

models.py

```

from django.db import models

# Create your models here.
from django.db.models import CASCADE

class ClientRegister_Model(models.Model):

    username = models.CharField(max_length=30)
    email = models.EmailField(max_length=30)
    password = models.CharField(max_length=10)
    phoneno = models.CharField(max_length=10)
    country = models.CharField(max_length=30)
    state = models.CharField(max_length=30)
    city = models.CharField(max_length=30)

class profile_identification_type(models.Model):
    prof_idno= models.CharField(max_length=3000)
    name= models.CharField(max_length=3000)
    screen_name= models.CharField(max_length=3000)
    statuses_count= models.CharField(max_length=3000)
    followers_count= models.CharField(max_length=3000)
    friends_count= models.CharField(max_length=3000)
    created_at= models.CharField(max_length=3000)
    location= models.CharField(max_length=3000)
    default_profile= models.CharField(max_length=3000)
    prf_image_url= models.CharField(max_length=3000)
    prf_banner_url= models.CharField(max_length=3000)
    prf_bgimg_https= models.CharField(max_length=3000)
    prf_text_color= models.CharField(max_length=3000)
    profile_image_url_https= models.CharField(max_length=3000)
    prf_bg_title= models.CharField(max_length=3000)
    profile_background_image_url= models.CharField(max_length=3000)
    description= models.CharField(max_length=3000)
    Prf_updated= models.CharField(max_length=3000)
    Prediction= models.CharField(max_length=3000)

class detection_ratio(models.Model):
    names = models.CharField(max_length=300)
    ratio = models.CharField(max_length=300)

class detection_accuracy(models.Model):
    names = models.CharField(max_length=300)
    ratio = models.CharField(max_length=300)

```

views.py

```

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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')

```

manage.py

```

#!/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()
```


5. SCREENSHOTS

5.SCREENSHOTS

In order to execute this project, firstly we need to turn ON Wamp server. To run this project, open the project folder using command prompt. Then, simply type “python manage.py runserver” command and click enter. This terminal will give https port number, we need to copy that https port number and paste it on chrome or any other browser. After all this process we will get web page as shown below.



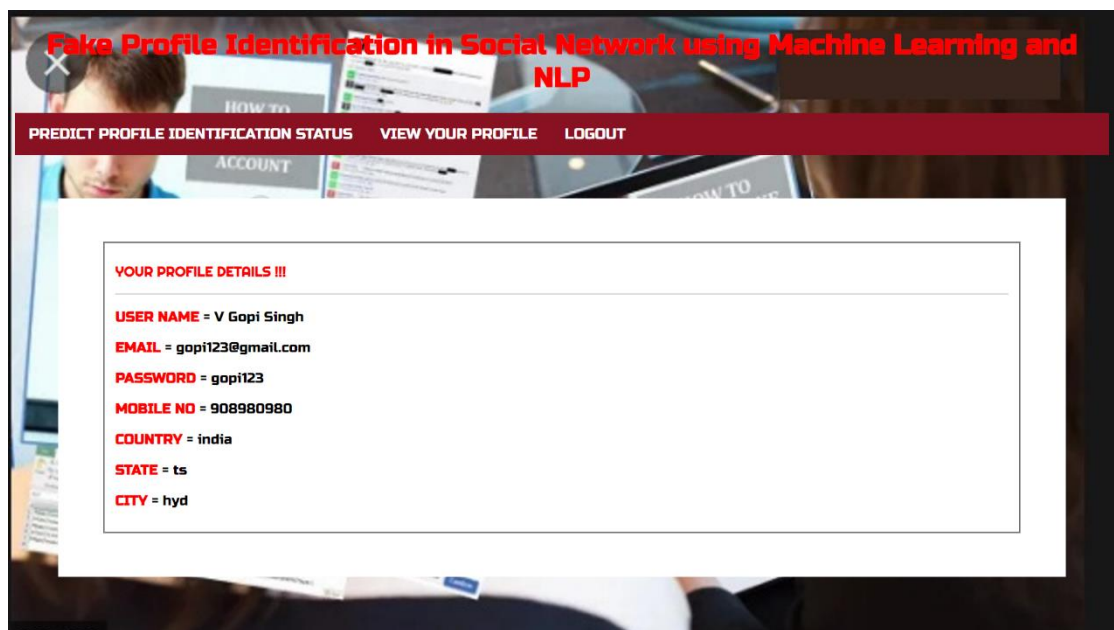
Screenshot 5.1: Remote User Interface.

In the above screen we can see the web page showing user login interface. Here in this page user can login if he/she registered before, otherwise he/she need to register first inorder to login.



Screenshot 5.2: Register page for remote user.

The above screen is opened after clicking on register button in “user login ” page. Here, in this page user can register as a remote user by entering the required details and the clicking on “sign_up” button.



Screenshot 5.3: User Profile Details After Registration.

The above screen will be seen after successful login with valid user name and password. On this page user can see his/her profile details and can predict profile identification status.

PREDICT PROFILE IDENTIFICATION STATUS TYPE!!!

ENTER ALL PROFILE ATTRIBUTE DETAILS HERE !!!

Enter Profile Idno	<input type="text"/>	Enter Name	<input type="text"/>
Enter Screen Name	<input type="text"/>	Enter Statuses Count	<input type="text"/>
Enter Followers Count	<input type="text"/>	Enter Friends Count	<input type="text"/>
Enter Created_at	<input type="text"/>	Enter Location	<input type="text"/>
Enter Default Profile	<input type="text"/>	Enter Profile Image url	<input type="text"/>
Enter Profile Banner url	<input type="text"/>	Enter Profile BG Image https	<input type="text"/>
Enter Profile Text Color	<input type="text"/>	Enter Profile Image url https	<input type="text"/>
Enter Profile BG Title	<input type="text"/>	Enter Profile background Image url	<input type="text"/>
Enter description	<input type="text"/>	Enter Profile Updated	<input type="text"/>

PROFILE TYPE PREDICTION STATUS:- -----Fake Profile -----

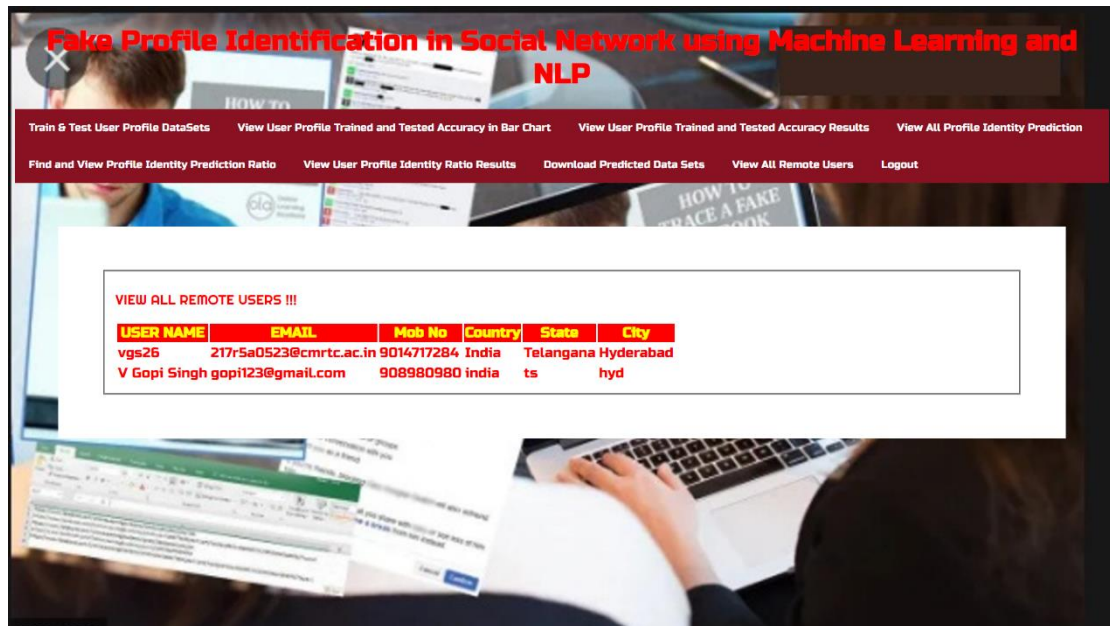
Screenshot 5.4: Predict Profile Whether Genuine Or Fake.

After clicking on “predict profile identification status” button user will get above screen. On this page user can predict whether profile is fake profile or genuine profile by entering all profile attribute details.



Screenshot 5.5: Service Provider login Interface.

This is the service provider login interface, using user name and password as “Admin” we can login.



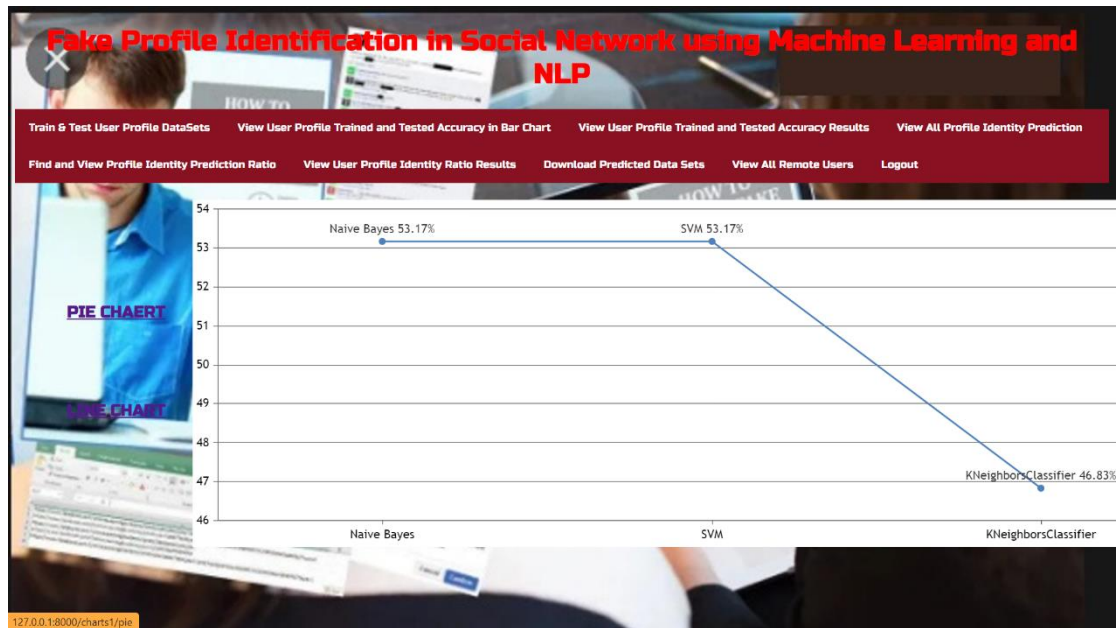
Screenshot 5.6: All Remote Users Details Who Registered.

The above page will be opened after logging in to service provider interface. In the above screen we can see all the remote users that are registered for predicting profile status.



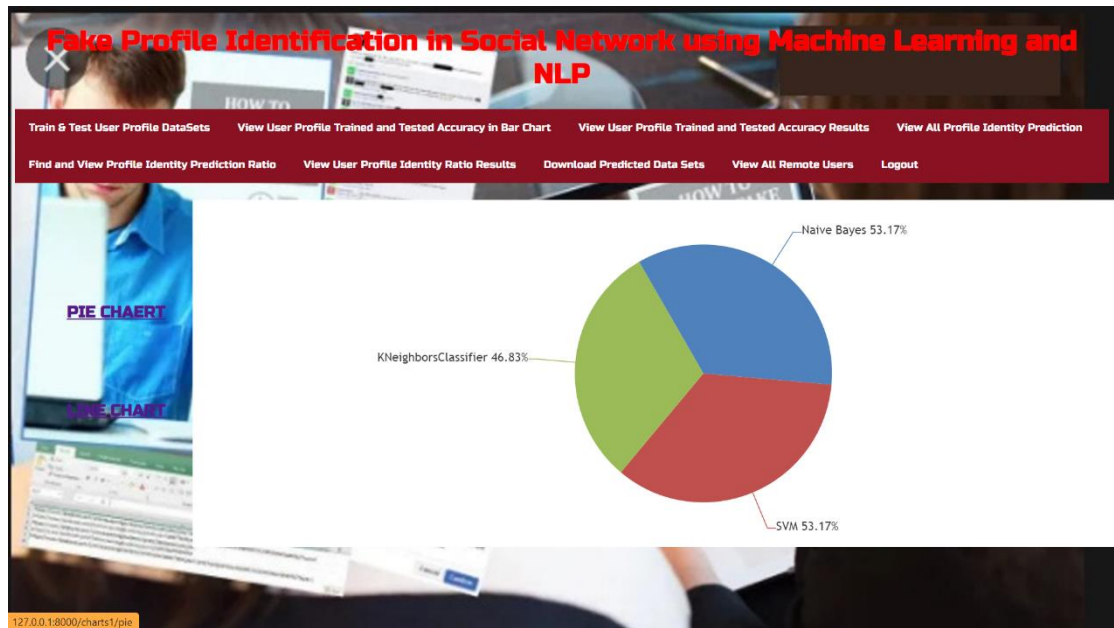
Screenshot 5.7: User Profile Trained And Test Accuracy In Bar Chart.

By clicking on “view user trained and tested accuracy in bar chart” button we can see the accuracy graph of algorithms used in this project for predicting profiles in bar chart showing the accuracy percentage.



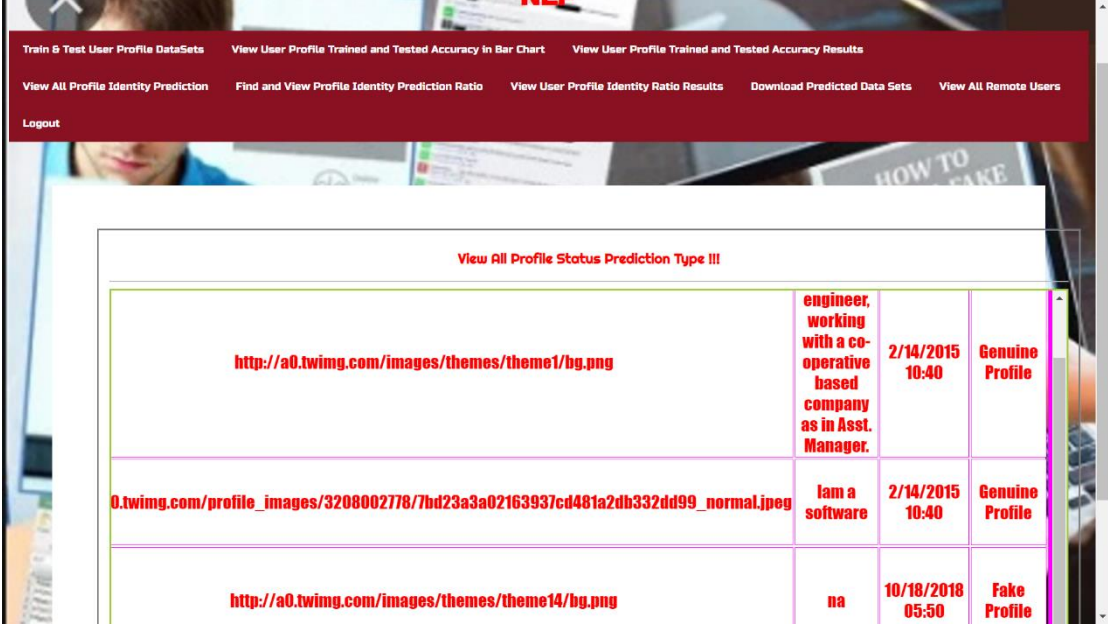
Screenshot 5.8: User Profile Trained And Test Accuracy In Line Chart.

In above screen, by clicking on “view user profile trained and tested accuracy” we can see line chart showing accuracy of algorithm used.



Screenshot 5.9: User Profile Trained And Test Accuracy In Pie Chart.

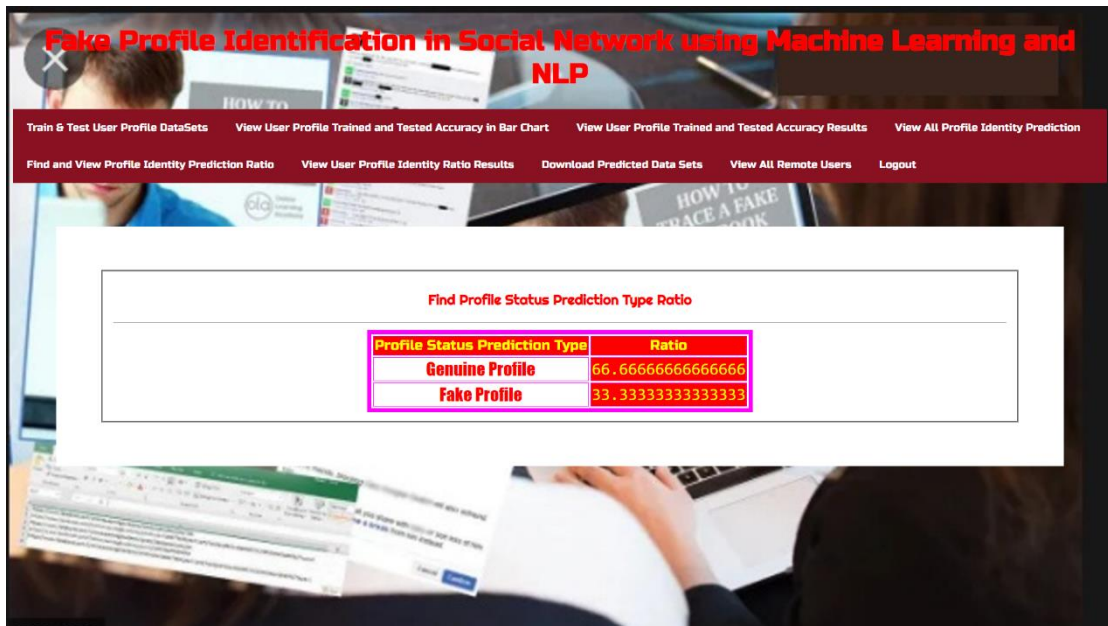
In the above screenshot we can see the another representation which displays accuracy of algorithm that is used in this project with pie chart.



View All Profile Status Prediction Type III			
http://a0.twimg.com/images/themes/theme1/bg.png	engineer, working with a co-operative based company as In Asst. Manager.	2/14/2015 10:40	Genuine Profile
0.twimg.com/profile_images/3208002778/7bd23a3a02163937cd481a2db332dd99_normal.jpeg	I am a software	2/14/2015 10:40	Genuine Profile
http://a0.twimg.com/images/themes/theme14/bg.png	na	10/18/2018 05:50	Fake Profile

Screenshot 5.10: All Profiles Status That Are Predicted.

By clicking on “view profile all identity prediction” we can see all the prediction that are made by all the remote users. In the above screen we can see the profile attribute along with the status wheather genuine profile or fake profile.

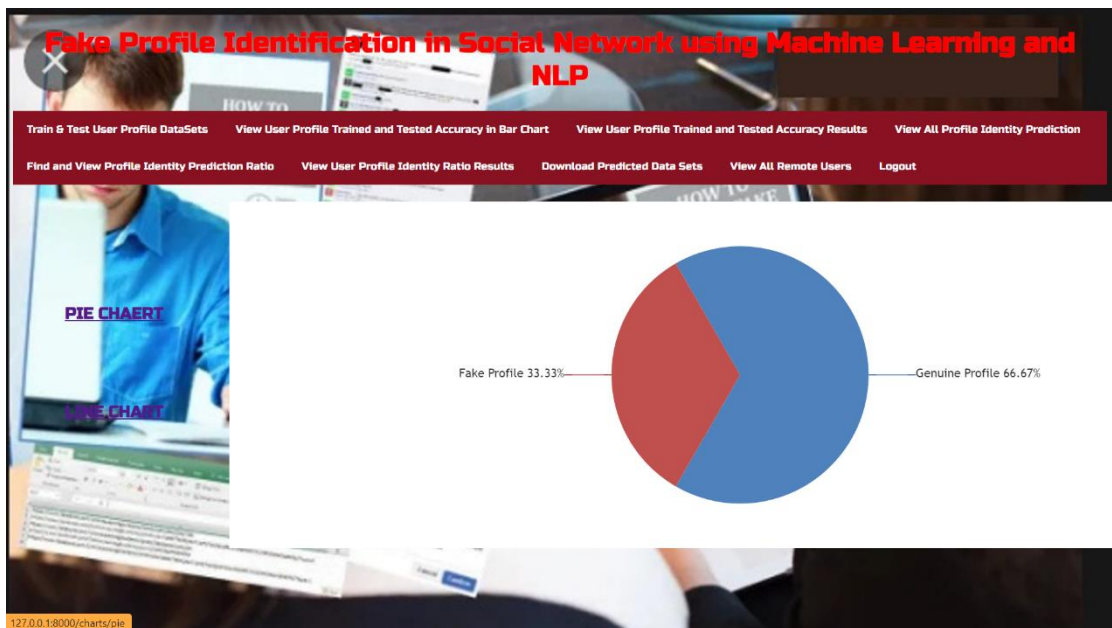


Screenshot 5.11: Profile Status Prediction Type Ratio.

The ratio of profile status prediction type is shown. In the above screen we see that genuine profiles are 66.66% and fake profiles are 33.33%. The values will change with the number of prediction results of identifying profile type.



Screenshot 5.12: Profile Status Prediction Type Ratio In Line Chart.



Screenshot 5.13: Profile Status Prediction Type Ratio In Pie Chart.

6. TESTING

6. TESTING

6.1 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, sub-assemblies, 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.

6.2 TYPES OF TESTING

6.2.1 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 tests ensure that each unique path of a business process performs accurately.

6.2.2 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 satisfactory, as shown by successful 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.

6.2.3 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 centred on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

6.2.3.1 SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

6.2.3.2 WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

6.2.3.3 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

UNIT TESTING

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

TEST STRATEGY AND APPROACH

Field testing will be performed manually and functional tests will be written in detail.

TEST OBJECTIVES

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

FEATURES TO BE TESTED

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Test Results:

All the test cases mentioned above passed successfully. No defects encountered.

ACCEPTANCE TESTING

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

7. CONCLUSION

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

In this project, we proposed machine learning algorithms along with natural language processing techniques. By using these techniques, we can easily detect the fake profiles from the social network sites. In this paper we took the Face book.Data set to identify the fake profiles. The NLP pre-processing techniques are used to analyze the dataset and machine learning algorithm such as SVM and Naïve Bayes are used to classify the profiles. These learning algorithms are improved the detection accuracy rate in this project.

7.2 FUTURE SCOPE

The future scope of Fake Profile Identification in Social Networks using Machine Learning (ML) and Natural Language Processing (NLP) involves advancements in deep learning, ensemble methods, and graph-based models for higher accuracy. Real-time processing capabilities will be crucial for instant detection and response to emerging fake profiles. Integration of multimodal data analysis (text, images, user behavior) will enhance system sophistication. Cross-platform integration across social networks will create a unified defense against fraud. Ethical considerations like user privacy and bias mitigation will continue to guide the development and deployment of these technologies, ensuring a trustworthy online environment.

8. BIBLIOGRAPHY

8. BIBLIOGRAPHY

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
- [2] Dr. S. Kannan, Vairaprakash Gurusamy, "Preprocessing Techniques for Text Mining", 05 March 2015.
- [3] Shalinda Adikari and Kaushik Dutta, Identifying Fake Profiles in LinkedIn, PACIS 2014 Proceedings, AISel
- [4] 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," in Computer Networks and Information Technology. (ICCNIT), 2011 International Conference on, July, pp. 35–390.
- [5] 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.

8.2 GITHUB LINK :-

<https://github.com/Vgs26/Fake-Profile-Identification-in-Social-Network-using-Machine-Learning-and-NLP->